カシオパソコン

PB-700活用法

BASICをやさしく解説し、実務をこなすためのパソコン活用決定版

監修●渡辺 茂　著者●長谷川 晃/繁野鎮雄　編集制作●技術評論社

取扱説明書

カシオ計算機株式会社

カシオパソコン

PB-700活用法

BASICをやさしく解説し、実務をこなすためのパソコン活用決定版

監修●渡辺 茂　著者●長谷川 晃/繁野鎮雄　編集制作●技術評論社

カシオ計算機株式会社

監修のことば

いわゆる“パソコンブーム”が高まり始めてから，現在までそれほどの時間は経っていませんが，にもかかわらず，その間各メーカーからさまざまな機能をもった多くのパソコンが世に送り出されました．

その中で，ポケコン，ハンドヘルドという，いわゆる中・小型のパソコン部門でも製品の開発が進められ，ひとつの分野を形成してきました．

カシオ計算機(株)では，昨年，1万円台というそれまでの常識を破ったポケットタイプのパソコンPB-100を発売し，多くのユーザ層をつかみました．今回，この製品シリーズで，より高い機能を備えた上位機種PB-700を発売することになりました．

本書は，このPB-700を有効に活用していただこうとまとめられたものです．

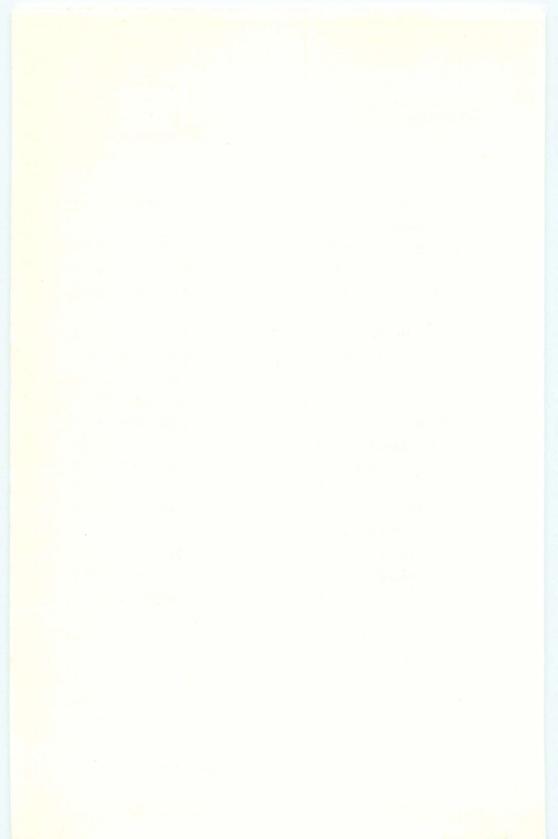
パソコンのプログラム言語であるBASICは，今では大変多くの命令語を持つようになってきましたが，ユーザの側からいえば，非常に便利に使うことができる反面，いざプログラミングの段になると，少々やっかいでもあります．

幸い，PB-700は，パソコン入門者にとって基本となる命令語約70種によって，計算，画面表示，Beep音，またオプションのカセットテープレコーダ，4色プロッタプリンタを使えますので，初心者の方がパソコンの使い方を学ぶ上では大変都合がよいということができます．

もちろん，本機のもつ多くの機能が，実務面やホビーの分野で大いに活用できることはいうまでもないでしょう．

そこで本書ではまず第1章と第2章でPB－700の基本機能と操作方法について説明した後，初心者の方々のためにBASICプログラミングの基本を第3章で詳しく解説しました．そして，第4章ではひとつひとつの命令語について，その機能と使い方を，サンプルプログラムを使いながらわかりやすくまとめました．そして最後に，ビジネス，グラフィックなどの分野別に，実用的なプログラムを何本かあげておきました．実際に皆さんの身の回りに応用されたらいかがかと思います．

是非皆さんが，本書によってパソコンをより深く理解され，身辺で大いに活用されることを望んでおります．

1983年8月　　渡辺　茂

はじめに

どこへでも手軽にカバンに入れて持ち運びでき，しかも卓上に設置するパソコンと同レベルの性能をもった“使える”パソコン……これがPB—700です．

なぜ“使える”のか．それにはいくつか理由がありますが，まず，RAM（ラム）容量が大きく，最大で16キロバイトにすることができることをあげるべきでしょう．RAM容量最大16キロバイトということは，ごく大雑把にいうと，文字数にして約15000字ぐらいを記憶させることができることを意味しています．これくらいあれば，個人が普通に使用する仕事のプログラムであれば，十分収容することができます．

RAMとは，ランダム・アクセス・メモリーの略で，ユーザーが自由に読み書きできる記憶装置のことで，ここにプログラムやデータを格納することになるのです．

さて，このRAMに，自由にデータを書きこんで，必要に応じてそのデータの加工を行ない，必要なデータのみを取り出すには，BASICと呼ばれるプログラム言語によって，最初にこのRAMに書き込んでおかねばなりません．ですからBASICのプログラムは，仕事の手順を書いたもの，ということになります．

コンピュータは，このプログラムなしには，ただの箱に過ぎないのです．そこで，本書は，これからBASIC言語を習得したいというユーザーにも十分PB—700のいろいろな機能を使いこなせるようなマニュアルとしての役目と，BASICの習得も容易に行なえる入門書としての役目の両方を満足できるように工夫して執筆してみました．

PB—700はオプションのカセットインタフェイス付ミニプロッタプリンタに接続することで，その機能が大きく拡大されます．これによる魅力は，何といってもミニプロッタプリンタの活用に尽きるでしょう．4色の美しいグラフや表などをす早く作成してくれます．本書はこの利用法にも触れていますが，こうしたすぐれた出力結果を印字してくれるのも最初に述べた，“使える”パソコンの理由であります．PB—700は使い込むほど手離せなくなり，愛着を感じさせてくれる機械だと思います．読者の一人でも多くの方がこうした愛着をPB—700に持って頂く糸口になれば，本書の役目を果たしたことになるでしょう．

1983年8月　　　　著者

目　次

第1章　基本操作

第2章　キーの基本操作と画面

第3章　BASICプログラム

CONTENTS

第4章　コマンド・リファレンス

目　次

CONTENTS

目次

このマニュアルに書いてあること

PB－700の特長は，たくさんあります．それは，パソコンの初心者にとって，短期間に習得することが大変なくらい，いろいろな機能が盛り込まれている．ということでしょう．いろいろな機能を一度に習得することは，なかなか大変なことだと思います．

原則的には，時間をかけてひとつひとつ理解し，自分のものにしていく以外に習得の方法はないのです．一般に，パソコンに対する習熟度は，そのキーに触れている時間に比例すると考えられます．

PB－700を正しく扱っていただくため，本機の概要と取り扱い上の基本的注意点を第1章で，また，BASICを学ぶ前の各キーの働きや表示画面について知っていただくため，第2章でこれらを解説してあります．

コンピューターの主要な機能は，いうまでもなく記憶することと計算することの2つに集約できます．PB－700にたくさんのデータを憶えさせ，いつでも必要に応じて必要なデータを自由にとり出したいというとき，BASIC言語を使用して，どのようにプログラムを作ればよいのでしょうか？　こうした疑問にお答えするため，第3章では，配列プログラムの作りかたを詳しく解説してあります．

また第3章は，BASIC の初心者の方がたのために，BASIC プログラミングの基本について説明しました．さらにこの章では，PB－700の大きな表示窓を活かすためのグラフィックプログラミングの基本についても触れ，PB－700の機能を拡張できるように，別売のカセットインタフェイス付ミニプロッタプリンタの活用法についても解説しました

すでに，BASICをマスターされている方がたは，必ずしも第3章を読む必要はないでしょう．むしろ必要なのは，第4章となります．この章は，PB－700のもつ多くのコマンドや関数について，その使用法をできるだけていねいに解説してあります．初心者の方は，この章のコマンドを最初からひとつずつ習得していくのは，効率的ではありません．構文上必要なBASICの基本コマンドは10個ぐらいですから，それらを中心に十分理解されるよう，選びだして学習することをお推めします．

第5章では，PB－700をすぐ実践に応用できるよう，代表的な用途のプログラム例を紹介してあります．これらを参考に，使いやすいようプログラムを手なおしして，PB－700を大いにそして有効に活用して頂きたいと思います．

PB-700と他のカシオパソコン

PB－700は，下の表のように，PBシリーズの中で，豊富なBASICコマンドをもった高級機に位置づけられ，表示窓も大きく，中・小規模の実務に十分使用できうるメモリー容量をもっています．

また，オプションのカセットインタフェイス付ミニプロッタプリンタも大きな魅力で，グラフやホビーにと，PB－700の機能を飛躍させてくれます．

機　種　名	RAM容量	特　長
FP－1100	メイン64Kバイト サブ　48Kバイト	・デュアルCPU．すぐれた計算精度（10進演算），倍々精度 ・豊富なグラフィック機能
FP－1000	メイン64Kバイト，サブ16Kバイト	・FP－1100のベースマシン
FP－200	最大32Kバイト	・A4サイズのハンドヘルド ・20桁，8行の表示パネル ・簡易表言語CETL搭載
PB－700	最大16Kバイト	・20桁，4行の表示パネル ・カセットインタフェイス付ミニプロッタプリンタ ・FP－200級の強力BASIC
PB－300/FX 802P	約2.3Kバイト	・プリンタ付
PB－200/FX 700P	約2.3Kバイト	・拡張メモリー付
PB－100	最大約2.3Kバイト	・手軽なBASIC入門機

（1983年9月現在）

第1章
基本操作

ご使用の前に　1-1

この計算機は，カシオ計算機の高度な電子技術と品質管理のもとで，厳重な検査工程を経て，皆様のお手もとに届けられています．本機を末ながくご愛用いただくために，次の点にご留意のうえ，お取り扱いください．

1 ご使用上の注意

— 本機は，精密な電子部品で構成されていますので，絶対に分解しないで下さい．また，投げたり落したり等のショックを与えたり，車の中・暖房器具の近く等の高温の場所に放置しないで下さい．また，湿気やホコリの多い所での使用や保管はさけて下さい．

気温が低いときは，表示の応答速度が遅くなったり，点灯しなくなることがありますが，通常の温度になると正常に戻ります．

— オプション接続用コネクターには，本機専用のカセットインタフェイス付ミニプロッタプリンタFA-10以外の機器を接続しないで下さい．

— 計算機のお手入れは，シンナー，ベンジンなどの揮発性液体をさけ，乾いた柔かい布，あるいは中性洗剤液に浸し固くしぼった布でおふき下さい．

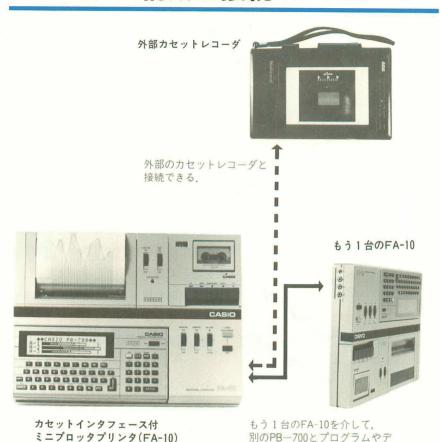
2 保証・アフターサービス

— 保証は，同封の保証書の内容によりますので，よくお読みのうえ，記入事項を確認して，大切に保管してください．

— 万一故障したときは，①お買い上げ店，②カシオ計算機サービスセンターのうち，ご都合のよい所へ，必ず保証書を添えて，ご持参またはご郵送ください．この場合は，故障内容を具体的にお知らせください．

— 修理依頼される前に，この説明書をもう一度お読みになると共に，電源状態および，プログラムミス，操作ミスがないかをよくお調べください．

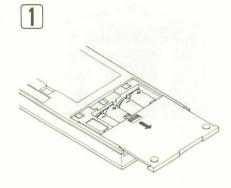
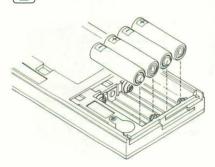
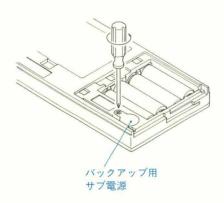
— ご不明の点やご質問，お問い合わせは，最終ページ記載のカシオ計算機へご連絡ください．

システム構成と接続　1—2

PB—700は，カセットインターフェイス付ミニプロッタプリンタFA-10を介して，さらに外部カセットレコーダーと接続できます．また，もう一台のFA—10を介して，別のPB—700とプログラムやデータの受け渡しを行なうことができます．

PB—700本体とFA—10との接続は，電源スイッチOFFで行ないます．

増設RAM OR—4の装着や取り外し後は，電源スイッチON後，**NEW ALL** ⏎の操作を行なって下さい．

電池の交換とリセット　1-3

1

3

■ 電池の入れかた

必ず電源スイッチをOFFにし，PB-700本体を裏がえし，電池ボックスのフタをスライドして下さい(1参照).

電池の交換時，プラスとマイナスを間違わないようご注意下さい．本体のスプリングの側に，電池のマイナスがくるようにセットします (2参照).

電池は，単３の乾電池４本です．交換する時は，古い電池と新しい電池を混ぜて使用しないで下さい．電池寿命を著しく短かくしてしまいます．

長期間使用しないときは，電池を抜きとったまま本体を保管して下さい．

■ 電池の交換時期

ブザーの音が小さくなったり，表示をしなくなったら，電池を交換して下さい．なお，電池の寿命に関しましては最後の「規格」をご参照下さい．

■ PB－700の電源の構成

PB－700の電源は，メイン電源と，RAMバックアップ用のサブ電源(3)に別れています．ですからメイン電源の交換時にも，またはサブ電源交換時にも，本体内のプログラムやデータが消失してしまうことはありません．

メインとサブの両方を同時に外した場合のみ，プログラムやデータが消失します．この場合は，電池交換後NEW ALL ⏎ の操作を実行して下さい．

1-3

■ 電池によるメモリーの保護について

リチウム電池のサブ電源は，メイン電池交換時のメモリーの保護のためにあります．

メイン電池を取りはずした状態で，4KBなら約10ヵ月，16KBなら約2ヶ月半，サブ電池は，RAMをバックアップします．

■ 電池交換時の注意

① 電池を交換するときは，必ずパワースイッチをOFFにして行なって下さい．

② メイン電池とサブ電池を同時にはずさないで下さい．

③ サブ電池は，PB―700を使用しない場合でも，2年に1回交換する必要があります．⊕と⊖を間違わないよう，よく確認して下さい．

④ 電池の寿命については，裏表紙内側の規格を参照して下さい．

⑤ とくにサブ電源のリチウム電池は，幼児の手のとどかないところに保管して下さい．万一飲み込んだ場合は，ただちに医師に相談して下さい．

⑥ 使用済の電池は，絶対に火中に投入しないで下さい．破裂することがあり危険です．

■ オートパワーオフ機能

電源スイッチONの状態で放置しておくと，約8分で自動的に電源OFFになります．これは，スイッチの切り忘れによるムダな電力消費を防ぐ自動節電機能で，この機能が働いた場合，BRKキーを押すことで，再び電源 ONの状態になります．なお，プログラム実行中（INPUT 文やPRINT 文等による停止中は除く）は，この機能は働きません．

■ 電圧低下検出機能

メイン電池が消耗してくると，表示が（徐々にではなく）パッと消えます．この時は，メイン電池を交換して下さい．

増設RAMパックのセット　1-4

PB―700のRAMを 増設しない場合は，4Kバイトです．別売の増設RAMパックOR-4を追加することで，最大RAMは，16Kバイトまで拡大できます．ひとつの増設RAMパックは4Kバイトです．

この装着法は，次の手順によります．

① 本体の電源スイッチをOFFにする．

② 裏がえしてRAMボックスのフタにある2個所のツメを同時に押えながら，フタをはずす．

③ 増設RAM OR-4のふちを持って，右からの所定の位置へのせる．このとき，OR-4の本体の端子に指などが触れないようにご注意下さい．RAMは，人体が帯びている静電気に弱いのです(1参照)．

④ 増設RAMの押さえ金具を軽く押しつけながら，金具のつめがみえなくなるまで2の矢印方向へ押す．

⑤ 増設RAMパックは，3の1～3の順に装着して下さい．1の位置をとばして，いきなり2の位置に装着すると，正しく機能しません．

⑥ 必要な数の増設RAMパックを同様に装着後，RAMボックスのフタをして，本体電源をONして下さい．

⑦ NEW ALL ⏎ の操作を実行します．なお，SHIFT S(SYSTEM) ⏎ で，RAM容量を確認できます．

1 電源スイッチOFF後……

①のように増設RAMを斜めにし後部を合わせ，②の前部をケース内に落す

2 押え金具をスライドさせる

増設RAMの押え金具を軽く押しつけながら③の矢印方向へ金具のつめがみえなくなるまで押す

3 増設RAMパックを入れる順番

パックの装着は必ず1，2，3の順に増設してください．1の位置をとばして2，3に装着しないでください．

各部の名称と働き　1-5

・**コントラスト調整ボリウム**……これをまわして，表示がもっとも見やすい濃度になるように調整して下さい．

・**アルファベットキー**……そのまま押すと，大文字のアルファベットが表示され，キャピタルシフトキーとアルファベットキーの同時押しで，小文字となります．また，シフトキーとの同時押しによって，キーの上のコマンドを表示させることができます（ワンキーコマンド）．

例

SYSTEM ———シフトモード
[S] ———ダイレクトモード

☞ 直接キーを押して入力できない記号や文字は，**CHR$**関数(212 ページ参照）を用います．

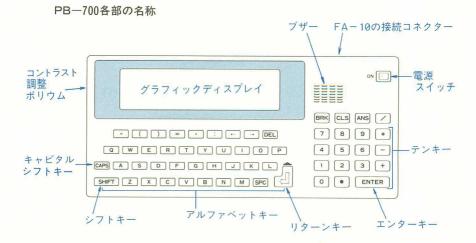
・**リターンキー(⏎)**…………BASICのコマンドやBASICプログラムの入出力，プログラムによるデータの入出力の実行命令を行ないます．

・**エンターキー([ENTER])**……電卓としてマニュアル計算を行なうとき，その実行の命令となります．

例　[1][2][/][4][ENTER]　→　**3**

・さらに詳細な各キーの使用法は，第2章編集用キー・特殊キーの機能の項を参照して下さい．

PB—700各部の名称

テストラン　1-6

PB-700の機能を，大雑把ですが，知っていただくためのデモンストレーションプログラムです．もしプログラムを正しく入力する自信がない場合には，2章，3章を読んでから，入力されるとよいでしょう．

```
 10 CLS
 20 LOCATE 3,1:PRINT "PB-700 TESTING"
 30 FOR A=31 TO 0 STEP -1
 40 DRAW(0,A)-(159,A)
 50 NEXT A
 60 LOCATE 5,1:PRINT "BEEP SOUND"
 70 FOR A=0 TO 9
 80 BEEP 1:BEEP 1:BEEP 0
 90 NEXT A
100 CLS
110 FOR A=33 TO 255
120 PRINT CHR$(A);
130 NEXT A
140 FOR A=0 TO 20
150 NEXT A
160 CLS
170 FOR A=1 TO 16
180 DRAW(A+2,16-A)-(A*3,16-A)-(A*3,15+
    A)-(A+2,15+A)-(A+2,16-A)
190 NEXT A
200 FOR A=0 TO 200
210 NEXT A
```

第2章
キーの基本操作と画面

ダイレクト・モードのキーの機能　2−1

ダイレクト・モードで各キーを入力すると，そのキーの頭にある文字または機能が入力されます．

A 〜 Z　英大文字

SPC（スペース）空白

"，=，(，)，'，:　それぞれの記号

Ø 〜 9　置数

•　小数点

+，−，*，/　四則記号

CLS（クリアス／クリーン）表示（画面）のクリア

DEL（デリート）1文字削除

BRK（ブレイク）実行の中止

ENTER（エンター）マニュアル演算（電卓のように）の実行

↵（リターン）プログラムの書き込みと実行

⇦，⇨　カーソルの左右移動

SHIFT（シフト）シフトモードの設定

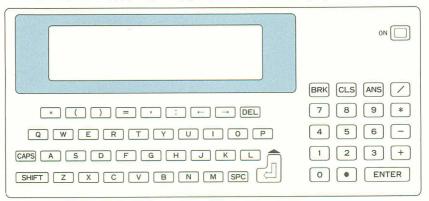
CAPS（キャピタルシフト）キャピタルモードの設定

ANS（アンサー）ひとつ前の計算結果を呼び出す

■1行の文字数

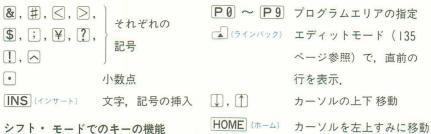
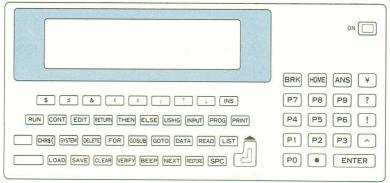
マニュアル計算における計算式，BASICにおける1行に書き込める文字数とも，最大79文字までです．

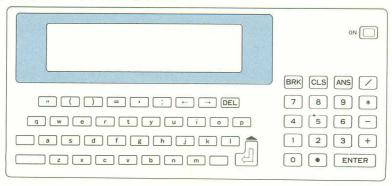
ダイレクト・モードでのキーの機能

（表示されている文字・記号・数字は全てワンタッチで機能します）

シフト・モードでのキーの機能　2-2

SHIFT キーとの同時押しにより有効となる機能です．各キーの上に赤字で表示してある文字，記号などが入力されます．

ワンキーコマンド26種類

キー	機能
&, #, <, >, $, ;, ¥, ?, !, ^	それぞれの記号
•	小数点
INS（インサート）	文字，記号の挿入
P0 ～ P9	プログラムエリアの指定
▲（ラインバック）	エディットモード（135ページ参照）で，直前の行を表示．
↓，↑	カーソルの上下移動
HOME（ホーム）	カーソルを左上すみに移動

シフト・モードでのキーの機能

CAPS　2-3

CAPS キーとの同時押しにより有効となる機能です．主にアルファベットの小文字を表示させるときに使います．

編集用キー，特殊キーの機能　2-4

(1) SPC ……(space) すべてのキーモードにおいて空白を出力します．

(2) HOME CLS ……(clear screen / home) 表示をクリアしてカーソルを左上端に移します．

SHIFT HOME CLS は，表示をそのままで，カーソルのみ文の書き始めに移します．

(3) INS DEL ……(delete/insert) 現在カーソルのある位置の文字や記号を削除し，カーソル以後の文字などをひとつずつ左に移動させます．

SHIFT INS DEL は現在カーソルのある位置を含めた右側の文字や記号を，ひとつずつ右側へ移動させて空白を挿入します．

(4) ON BRK ……(break) 演算やプログラムの実行を中止します．あるいは，オート・パワー・オフ時の電源投入に使います．

(5) ↑ ← ，↓ → ……カーソルを左右，上下に移動させます．リピート機能は左右の移動のみです．

(6) ENTER …(enter) マニュアル演算を実行します．INPUT 文でキー入力待ちをしている場合には，⏎ と同様に機能します．代入文，コマンド，ステートメントは，ENTER で実行させることはできません．

(7) STOP ANS ……(answer)直前に実行された演算およびPRINT，LPRINT 文により出力された数値の結果を記憶します．

（例） 3.4 ＊ 5 ENTER → 17

5.8 ＊ 3 － ANS ENTER → 0.4

また，STOP ANS は，プログラム実行中，プログラム停止機能として働き，CONT文によりプログラムを再開できます．

(8) ▲ ⏎ ……(return/line back)プログラムの書き込みおよびコマンドの実行をします．マニュアル演算の実行はできません．SHIFT ▲⏎ によって，EDITモード(135 ページ参照) において現在表示されている行の直前の行を表示させることができます．

(9) SHIFT P0 0 ～ SHIFT P9 9 　プログラムエリアナンバーを指定して，先頭行から実行します．

演算機能　2-5

■演算精度と機能

内部数値演算は，すべて仮数部12桁(＋指数部2桁)で行ないます．10進演算ですから，高精度の演算を行なうことができます．

マニュアル演算は，ENTER キーで実行します．

演算結果は，仮数部10桁（＋指数部2桁）で表示します．この際，仮数部11桁目を四捨五入します．

■演算子の働き

^　　べき乗

＋，－　加算，減算

＊，／　乗算，除算

MOD(モッド)剰余(余り演算：扱う数値が小数部を含む場合は，小数部を切り捨てて演算)．演算範囲は，次の通りです．

①　0による除算はMAエラー

②　オーバーフローをしたとき(演算範囲を越えたとき)はOVエラー．

③　べき乗の演算範囲

$0 \wedge 0$ ⟶ MA エラー

$(\pm x) \wedge 0$ ⟶ 1

$0 \wedge y$ ⟶ 0

$0 \wedge (-y)$ ⟶ MA エラー

$(-x) \wedge (\pm y)$ ⟶ yが整数のみ可．他はMAエラー．

☞ x，yは共に正数

■演算の優先順位

演算は，次の順で実行します．

1．カッコで囲まれた中
2．関数
3．べき乗(^)
4．符号（＋，－）
5．＊，／
6．MOD
7．＋，－
8．関係演算子（=，>，<など）

■関係演算子の使い方

関係演算子は，IF 文（184ページ参照）においてのみ使用できます．

＝	等しい
<>，><	等しくない
<	小さい
>	大きい
＝>，>＝	大きいか等しい
＝<，<＝	小さいか等しい

【例】

A＋B<>　0　……A＋Bの答えが0と等しくない．

A$< > "Y"　……A＄の内容が，「Y」ではない．

A$＝CHR$(192)＋CHR$(197)＋CHR$(182)

……A＄が「タナカ」に等しい

CHR$(67)>CHR$(N)　……キャラクタコード表で，CHR$(67)

つまりCがCHR$(N)より大きい

■変数による演算の考え方

変数の内容は，ENTERキーで確認することができます．

【例】　A ENTER ⟶ 0

変数に数値を入れた場合，変数の内容は次のようになります．

単精度：仮数部13桁目以後切り捨て（12桁）

（単精度とは，普通の状態の計算精度のことです）．

半精度：仮数部６桁目以後切り捨て（５桁）

（半精度とは，配列変数！の指定により行ない，５桁の数値となります．配列変数においてのみ使用できます）．（数値配列変数の練習62ページ参照）．

☞**半精度計算は，こんなに便利．**技術計算などを除けば，計算は，５桁あれば十分という場合が少なくありません．例えば，成績，構成比(％)などや製品番号，単価，数量なども，５桁以内で問題なければ，半精度を使用すると，使用メモリーが単精度の半分ですみ，より多くのデータをRAMに置くことができます．

使用するメモリーは，通常(単精度)で，１数値変数当たり８バイト必要ですが，半精度にすると，これが4バイトですむのです．

演算式の記述法

【例】　●書　式

1. $\frac{X+Y}{2}$ ⟶ (X＋Y)/2
2. $X^2+2XY+Y^2$ ⟶ X^2＋2＊X＊Y＋Y^2
3. $-Y^2$ ⟶ －Y^2
4. $(-Y)^2$ ⟶ (－Y)^2
5. $(X^Y)^2$ ⟶ X^Y^2
6. X^{Y^2} ⟶ X^(Y^2)
7. $\frac{X}{Y}$ の余り ⟶ X MOD Y

●実際の演算例

1. 0.5 ^ 0 ⟶ 1
2. −0.5 ^ 0 ⟶ −1
3. (−0.5) ^ 0 ⟶ 1
4. 0.5 ^ 2 ⟶ 0.25
5. 0.5 ^ −2 ⟶ 4
6. (−0.5) ^ −2 ⟶ 4
7. 0.5 ^ 0.5 ⟶ 0.7071067812
8. (−0.5) ^ 0.5 ⟶ MA error
9. 2 ^ −0.5 ⟶ 0.7071067812
10. (−2) ^ −0.5 ⟶ MA error
11. 10 MOD 6 ⟶ 4
12. −10 MOD 6 ⟶ −4
13. 10 MOD −6 ⟶ 4
14. −10 MOD −6 ⟶ −4

変数　2-6

■ 変数の種類

PB－700の変数の種類は，次のようになっています．

①数値変数――数値固定変数（12桁まで）
数値登録変数（12桁まで）
数値配列変数（半精度数値配列＝5桁，単精度数値配列＝12桁まで）
※上記桁数は内部演算桁数です．
文字固定変数（7文字まで）
文字登録変数（16文字まで）
文字配列変数（1～79文字まで，任意の文字数を指定できる．指定しない場合は，自動的に16文字に設定される）

〔例〕 配列変数で，使用できる文字数を設定する場合

書式　DIM A$(9,9)*79

意味　A$(0,0)，A$(0,1)……A$(9,9)の変数には，それぞれ最大79文字まで格納することができる．

■ 固定変数（A～Z又はA$～Z$）

数値や文字を記憶するメモリーは，A～ZまたはA$～Z$の計26あります．ひとつのプログラム中で，同一名の数値固定変数と文字固定変数の両方を同時に使用することはできません．もし同一変数名を使用した場合，UVエラーとなります．

〔誤った使い方〕

10　PRINT A；A$ → UVエラー

■ 登録変数

固定変数の他に英大文字および数値で構成される2文字の変数名を使うことができます．3文字以上の変数名を定義すると，実行時にSNエラーとなります．

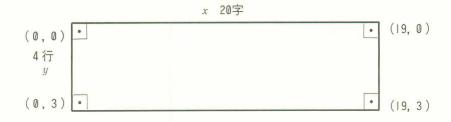
【例】AB，X1，Y1，X2，Y2，AZ$，AA$，B1$，Z9$

・変数名は，先頭が英大文字でなければなりません．

・予約語（IF, TO, PI など）は変数名とすることはできません．
・数値登録変数(AB, X1 など)には，仮数部12桁＋指数部２桁を格納できます．
・文字登録変数には，最大16文字を格納できます．
・登録変数の使用できる個数は，〔40－(配列変数名の個数)〕です．つまり，配列変数名の個数と合わせて40個ということです．40個を越した場合は，VA エラーとなり，実行を中断しますから，CLEAR（160ページ参照）もしくは ERASE（171ページ参照）で変数名を削除して下さい．
LIST V を実行することにより，登録されている変数名を呼び出すことができます．なお，数値登録変数は８バイト，文字登録変数は17バイト分使用します．

表示画面の知識　2-7

キャラクター座標

画面(LCD)は，横20文字，たて４行の文字(キャラクター)が入ります．文字の位置の表わしかたは，次の座標に従って，LOCATE文により行ないます．

LOCATE（x, y）……………LOCATE 参照
これにより表示できるキャラクターは，巻末キャラクター表の 221 種類です．

グラフィック座標

ドット単位で表示を指定したい場合は，次の座標によります．ドットの位置の表わしかたは，DRAWまたはDRAWC文により行ない，点や直線を描いたり消すことができます．

DRAW（x, y）……………DRAW, DRAWC参照

また, POINT関数により, グラフィック座標上の点が点燈しているか否かを調べることができます.

POINT（x, y）……………POINT参照

変数の使用バイト数　2-8

固定変数を除くすべての変数は, プログラム実行上でデータが代入されると, 残りのRAM容量が減少していきます. それぞれの1変数当たりの必要なバイト数は, 次のようになります. なお, A～Zの固定変数は, 自由に使用できるRAMエリアとは別に確保されていますから, バイト計算の必要はありません.

登録変数		配列変数	
変　数	使用バイト数	変　数	使用バイト数
数値変数	8　バイト	数値変数（半精度）	4　バイト
文字変数	17　バイト	数値変数（単精度）	8　バイト
		文字変数	2～80バイト(1～79文字)

第3章
BASICプログラミング

初めてのBASIC　3-1

BASIC（ベーシック）という言葉は，すでに皆さんはたびたび目や耳に触れられていることと思います．

BASICは，Beginner's All-purpose Symbolic Instruction Code（初心者向多目的記号命令）の頭文字をつなげて作られた名前である，といわれていますが，実際にはその名前の通り，Basicな言語という解釈の方が正しいともいわれています．

どちらの解釈をとるにしても，BASICは文字通り初心者にも習得の容易な，コンピュータの入門に適した高級プログラム言語です．ちなみに，日常の言葉に近い用語でプログラムができる言語を，高級プログラム言語といいます．

高級プログラム言語には，BASICの他にも，FORTRAN，COBOL，PASCAL，PL/Iなど数えきれないほど多くの種類があります．BASICも，始めは大型の計算機で使うために，1964年，米国の大学で生まれた言語でした．

たくさんの言語のひとつにすぎなかったBASICは，人間とコンピュータが対話しながらプログラムを作っていくことができるという便利な特徴を認められ，他の言語をさしおいていまやパソコンといえばBASICマシンといわれるほどに普及してきました．

BASICのよさは，何といってもこの対話形式とよばれるプログラムの作り方の方式にあります．いろいろ試しながらプログラムを作っていくことができるのです．

どこにでも携帯できるPB−700をお持ちのあなたならば，毎日少しずつ機械をいじっているうちに，必ずBASICでプログラムを作ることができるようになるでしょう．

キーに触れて下さい　3-2

PB －700は，大量のデータを処理し，複雑な数値計算をすることのできるコンピュータですが，プログラムを使わないマニュアル計算をすることもできます．

まずPB－700に慣れるために，ごく簡単なところからスタートしましょう．

電源スイッチをONにして下さい．画面には，

Ready　P0　（プログラムエリア0番が指定されています）

が表示されます．PB－700を電卓として使うときは，主として本体右側の数字キーを使うことになります．この部分のキーをテンキーと呼びます．テンキーを普通の電卓と比べると，[×][÷]キーそして[=]キーがありません．左側のキーには[=]キーがありますが，これは電卓の[=]キーの代りにはなりません．

[×][÷]キーの代りに，それぞれ[*][/]のキーを使います．[=]キーの役目は，[ENTER]キーなのです．

それでは，1＋2をやってみましょう．[1][+][2]と入力して下さい．

```
Ready  P0
1+2_
   └カーソル
```

表示は，このようになりましたか．もし，まちがいがあったら，[←][→]のキーを使って，＿（カーソル）をその個所へ移動させ，正しく入力しなおします．

次に，[ENTER]キーを押すと，答えが次のように表示されます．

```
1+2
  3
_
```

PB－700には，＋，－，＊，／の四則計算以外にも，ベキ乗，三角関数，逆三角関数，対数など数多くの関数が用意されていますから，もっと複雑な計算をすることもできます．複雑な式を計算する場合は，多少式を変形する必要の生じることもあります．

〔数式表現〕		〔入力する形〕
5×6÷2	➡	5*6／2 ENTER
6.5^2	➡	6.5^2 ENTER
$\frac{\text{SIN } 30^\circ + \text{COS } 60^\circ}{\text{TAN } 45^\circ}$	➡	(SIN 30 + COS 60)／TAN 45 ENTER

変数と代入　3-3

それでは，もうひとつ計算を行なってみましょう．

$500000 \times (1+0.07)^{10}$ ，$800000 \times (1+0.07)^{10}$

これは，複利計算による10年後の元利合計額を計算する式ですが，この2つの計算を行なうのに最も簡単な方法はなんでしょうか．それは，1回入力した計算式をくり返し部分的に使用するのですから，その繰り返し部分をメモリーにしまっておいて，必要のつどそれを呼び出して使用することです．そこで，

A＝(1＋0.07)^10 ↵

としておいて，前の2つの式を，

A＊500000 ENTER，A＊800000 ENTER

とした方が，ずっと楽に計算できます．

ここで，Aには $(1+0.07)^{10}$ の値がしまっておかれたことになるのですが，このAを，プログラム上では，変数と呼びます．

Aという変数に，ある数値をしまっておくには，次の操作を行ないます．

A ＝ 176 ↵　（Aに176を代入せよ）
（左辺）（右辺）

左辺Aに右辺176をしまうことを，代入する，といいます．ですから，この例でいえば，Aに176を代入する，ということになります．ここで，代入するという命令は，＝が行なっているのです．

それでは，Aに176が入ったか，確認してみます．A ENTER と操作して下

さい．これは，変数Aの中身を表示しなさい，という意味です．176が表示されましたか．

この点は，BASIC を理解する上で大変重要ですから，是非正しく理解しておいて下さい．

＝は代入せよ，という命令であって，等しい（イコール）という数学的意味ではないのです（IF 文を除く）．例えば，

A＝A＋1 ↵　　　（AにA＋1の値を代入せよ）

は，AにA＋1の値を代入せよ，という意味になります．ですから，いまAという変数に176が入っていれば，この式をコンピュータが実行すると，変数Aには177が代入されるのです．確認のため，A ENTER と操作して下さい．

表示は次のようになりましたか．

```
A
 177
_（カーソル）
```

■ ENTER と ↵ のちがい

さて，ここでもうひとつ大切な注意があります．それは，ENTER キーと ↵ キーの役割のちがいです．前述の操作で，A＝176は ↵ キーで，Aの値を表示させるのは ENTER キーだった点に注意して下さい．

つまり，ENTER キーは，電卓のような使い方，例えば答を表示しなさい，というときに使用します．これを，マニュアル計算といいます．

一方，↵（リターン）キーは，Aに176を代入するというようにBASICプログラム上の命令を実行するときに使用します．例えば，プログラム入力を実行せよ，プログラムのある個所を訂正せよ，BASIC のいろいろな命令を実行せよ，というようなときに使用します．

変数を使う練習　3-4

変数の名前には，A～Zのようにアルファベット１文字のものや，AAやN1のようにアルファベット＋１文字の２文字にして使用することができます．前項のように数値を代入された変数は，計算式中で自由に使うことができます．

では，練習をしましょう．

A＝36　↵
B＝12　↵

と操作して下さい．Aに36，Bに12を代入したことになります．次に，

A＋B　[ENTER]

マニュアル計算の「答を表示せよ」は[ENTER]キーです．48が表示されましたか．では次です．

A－B　[ENTER]　➡　24
A＊B　[ENTER]　➡　432
(4＋A)＊LOG B　[ENTER]　➡　99.39626599

このように，変数を自由に使いこなせるようになれば，相当複雑な計算を行なうことができます．しかし，それでも手順を追って計算したいときには，やはり上のやり方だけでは限界があります．そこで，いよいよプログラムを作ることになるのですが，変数を自由に使えるようになっていれば，プログラムを作るのはそれほど難しいことではありません．

■プログラムエリア

PB－700にいろいろなプログラムを入れておいて，必要に応じて選択して使用したいというとき，P０～P9の10個のプログラムエリアにそれぞれ異なるプログラムを入れておくことができます．電源ONで，「Ready P０」の表示が出ますが，このPの後の数字が，そのエリアを示しています．操作中現在どのエリアかを知りたければ，[BRK]キーを押します．

エリアの変更は，例えばP１のエリアへ移りたければ，

PROG　1　↵　または，[SHIFT] [PROG/0] [1] ↵ です．

プログラムを入力する練習　3-5

プログラムを理解する前に，プログラムを正しく入力できなければなりません．最初は，どこにどのキーがあるかを知るまでが大変ですが，すぐ慣れます．アルファベットキーの配列は，JIS で定められたもので，英文タイプライターに準じています．それでは，次の操作後，プログラムを入力して下さい．

まず，

SHIFT PROG/O 0 ↵ ➡ プログラムエリアをＰ０にする

N E W ↵ ➡ Ｐ０にあるプログラムを消す

CLS ➡ 画面をクリアする

〔プログラム〕	〔キー操作〕
10 CLEAR	1 0 SHIFT CLEAR/C ↵
20 A=A+1	2 0 A = A + 1 ↵
30 LOCATE 7,2	3 0 L O C A T E 7 , 2 ↵
40 PRINT A	4 0 SHIFT PRINT/P A ↵
50 GOTO 20	5 0 SHIFT GOTO/H 2 0 ↵

➡ └┘は，２つのキーの同時押しです．（21ページ参照）

入力が終ったら，CLS SHIFT RUN/Q ↵ または，SHIFT P0/0 を押して下さい．もし正しく入力されていれば，画面の中央に，１，２，３……と数字が高速で表示されます．

もし，SN error Ｐ０－行番号の表示が出たら，入力ミスがありましたので，指定された行番号を訂正（デバッグといいます）しなければなりません．その方法は，

SHIFT EDIT/E 行番号 ↵

です．この操作で，指定した行番号とプログラムが表示されますから，カーソルを移動させ，訂正し，↵ キーを押します．ここで，次の行番号が表示されますから，訂正する必要がなければ，BRK キーを押し，再び SHIFT RUN/Q ↵ と押して下さい．

プログラムの訂正は，エディット（EDIT）モードにする，ということを覚えておいて下さい．

BASIC プログラムの基本【1】　3-6

それではプログラミングに挑戦してみましょう．

右の図のような正方形の面積を求めるプログラムを作ってみます．

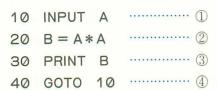
プログラムを作る手順は，

① 一辺Aはいくつにしますか，とたずね，入力してもらう．

② 入力された数値×入力された数値を計算する．

③ 計算した数値を表示する．

④ ①に戻る．

```
10  INPUT  A     ……………… ①
20  B = A*A      ……………… ②
30  PRINT  B     ……………… ③
40  GOTO  10     ……………… ④
```

このプログラムを，次の手順で入力してみましょう．文字の通り，正確にキーを押して下さい．

電源 ON	………Ready　P0の表示が出ます．
NEW [↵]	………[↵]は，アルファベットキーの右下にある矢印のキーです．
10 [SHIFT] [INPUT/I] A [↵]	……[SHIFT]と[INPUT/I]は同時に押します．
20 B = A * A [↵]	……*は，テンキーの所にあります．
30 [SHIFT] [PRINT/P] B [↵]	……[SHIFT]と[PRINT/P]は同時に押します．
40 [SHIFT] [GOTO/H] 10 [↵]	……[SHIFT]と[GOTO/H]は同時に押します．

では，このプログラムを実行させてみましょう．次の手順です．

[CLS] [SHIFT] [RUN/Q] [↵] または [SHIFT] [P0/0] と押して下さい．

すると，?_と表示されます．_は，カーソルと呼びます．ここで，

8.5　[ENTER]

と入力して下さい．

すると，次の表示が出ます．もし，この表示が出なかったら，[SHIFT] [EDIT E] [↵] で，プログラムの入力ミスがないか点検して下さい．Oと0，1とIは間違えないよう，要注意です．

```
RUN
?8.5              ………辺の値を8.5とする
  72.25           ………面積は，72.25です
?_ (カーソル)     ………辺の値をいくつにしますか？
```

このプログラムの実行を確認したら，プログラムについて考えてみましょう．

```
10 INPUT A
   入力しなさい Aに
```

10は行番号といい，プログラムの実行順を示します．ここでは10ずつふやしていますが，詳しくは後で述べます．INPUTは入力するという意味です．つまり辺の値をいくつにするか，?を表示させて，それに対する入力があるまで待つのです．入力があったら，それを数値変数にしまいなさい，という命令になります．

```
20 B=A*A        …………A*Aの結果をBに代入しなさい
```

20行は面積計算です．10行で入力された数値をかけ合わせ，つまり正方形の面積をBに代入します．

```
30 PRINT B
   表示しなさい Bを
```

30行は，面積の表示です．「表示しなさい」という命令には，PRINT を使うのです．Bの内容を表示しなさい，という命令になります．

40 GOTO 10

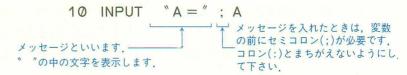
行きなさい 10行へ

GOTOは，「～へ行け」という命令です．この後には，行くべき行の番号を書くのです．

以上で，このプログラムで使用されている命令 INPUT，PRINT，GOTOの基本が分りました．でも，プログラムを実行してみて分る通り，？で何を入力したらよいか，また，計算結果がいったい「何の」計算結果なのかがよく分らず，大変不親切なプログラムです．そこで，少しプログラムに肉付けしてみましょう．その手順は次の通りです．

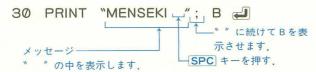
これで10行のプログラムが表示されますから，← キーを押してカーソルをAの所へ移動させ，次の操作で変更して下さい．

〈キー操作〉 "A =" SHIFT ; A

変更結果は次の通りです．

つぎに20行が表示されますが，変更なしなので を押して下さい．

つぎに，30行が表示されます．これをカーソル移動して，次の変更操作をして下さい．

〈キー操作〉 "MENSEKI SPC " SHIFT ; B

変更結果は次の通りです．

ここで，40行が表示されますが，この行は変更しないので，そのまま ⏎ キーを押して下さい．
それでは実行させてみましょう．

CLS　SHIFT　RUN/Q　⏎　………SHIFT　P0/Ø　でもよい．

どうでしょうか．こんどは，「A＝？」と聞いてくるし，計算結果も「MENS EKI 72.25」と表示してくれますから，前と比べて，ずっとよいプログラムとなりました．

BASICプログラムの基本〔2〕　3-7

基本となる命令を使ったプログラムを入れながら，BASIC の基本についてさらに学んでいきます．使うプログラムは次のもので，ある指定した数値の0～200までの倍数を出力します．

```
 10 REM BAISUU ⏎
 20 A=0 ⏎
 30 INPUT "BAISUU WA";N ⏎
 40 A=A+1 ⏎
 50 B=N*A ⏎
 60 IF B>200 THEN 100 ⏎
 70 PRINT B; ⏎
 80 INPUT " OK";C$ ⏎
 90 GOTO 40 ⏎
100 END ⏎
```

入力を終了したら，SHIFT キーを押しながら EDIT キーを押して下さい．次に ⏎ で入力したプログラムが最初から表示されますから，もし間違っていたら，カーソルを移動して訂正して下さい．下のような表示が出ましたか．

10 REM BAISUU

以下，BASIC プログラムの基本構成を知って頂くため，このプログラムを解説していきますので，説明文に従って，その通り ⏎ キーを押して下さい．

〔1〕

205ページ参照

BASIC のプログラムは，行番号から始まり，次にコマンド（プログラム命令または命令の一部），次いで変数または変数を使用した式で構成されます．この10行では，REM 文つまり注釈命令文なので変数はなく，代りにラベルと呼ばれる注釈がついています．このプログラムが倍数のプログラムであることを示しています．いうなれば見出しのようなものです．REM の後の文は実行されません．では，⏎ キーを押して下さい．

〔2〕

20 A＝0

行番号　数値変数（固定変数）

20行では，プログラム開始のとき，A という変数に 0 を代入しています．これを，変数の初期設定と呼びます．

PB−700 では，行番号は 1～9999まで，任意につけることができますが，プログラムの実行は，行番号の若い順に行ないますから，それを考えて行番号を割りつけます．

行番号は，10行おきにつけると見やすいプログラムとなりますし，後の変更もしやすくなります．

さて，20行は次のように表現しても同じことです．

20 LET A＝0

193ページ参照

LET とは，代入命令ですが，省略しても差しつかえありません．では，⏎ キーを押して下さい．

〔3〕

30 INPUT "BAISUU WA";N

行番号　入力命令　メッセージ文　変数

187ページ参照

INPUT は 入力命令で，キーからの数字や文字などの入力がない限り，次の行へ実行が移りません．入力された数字や文字を，メッセージ文の次の変数に代入し，次の行へ実行が移ります．

メッセージ文はなくてもかまいませんが，入れた場合には何を入力するべきかを表示させることができます．

INPUT は，「；」＋ 変数で，メッセージ文の後に「？」を表示します．変数の前の「；」を「，」にすると，「？」は表示されなくなります．

では，⏎ と押して下さい．

〔4〕

40　A＝A＋1

変数Aは，このプログラムの加算回数を計算するカウンターです．最初にこの40行を実行すると，20行の初期設定で，Aには0が代入されていますから，「AにA＋1を代入する」というこの命令は，A＝0＋1つまりAには1が代入されます．さて，Aが1の状態で，また40行を実行すると，次には，A＝1＋1　つまりAには2が代入されます．

このように40行の実行回数に応じて，Aには1が加算されていくのです．

40行の実行	1回目	A＝0＋1	（Aは1）	Aに，前のAの値に1をたした値を代入する．
	2回目	A＝1＋1	（Aは2）	
	3回目	A＝2＋1	（Aは3）	
		⋮		

では，⏎ キーを押して下さい．

〔5〕

50　B＝N＊A

40行と同じ代入文です．N＊Aの値を数値変数Bに代入する，という命令です．この行の最初の実行のとき，40行の実行によって，Aには1が代入されています．また，Nには30行の INPUT の実行によって任意の数値が代入されていますから，

B＝17×1　➡ Bには17が代入される

☞ 次の繰り返しの50行の実行によって，Aには，2回目は2が代入されるので，

```
2回目    B = 17 × 2
3回目    B = 17 × 3
            ⋮
```

数値変数Bには，Nの倍数が，つぎつぎと代入される．

ここまでは，理解できましたか．ある行の繰り返しの実行，とここでいっているのは，このプログラムが，40行と50行の計算を何回も繰り返しやらせるようになっているからです．その繰り返しの命令は，もう少し後で説明します．では，代入文が理解できましたら，次の BASIC 命令を学びましょう．⏎ キーを押して下さい．

〔6〕

```
60 IF B>200 THEN 100
```

IF／THEN：判定文　B>200：条件式　100：分岐先行番号

もし(IF)，Bの値が200よりも大きければ(B>200)，100行へとべ，という条件文です．つまり，このプログラムでは，50行を繰り返し実行していくと，Bの値が12回目で204となり，B＞200の式が成立し，100行へジャンプ（分岐）します．もし，Bの値が200に等しいか，小さければ，ジャンプせず，次の行番号へ実行が移ります．

IF 式 THEN 行番号　184ページ参照

式：この式が成立するならば，指定の行番号へジャンプする

この命令を利用したのです．では，⏎ キーを押して下さい．

〔7〕

```
70 PRINT B;
```

195ページ参照

PRINT：画面表示命令　B：数値変数

PRINTは，「表示しなさい」という命令です．このプログラムでいえば，数値変数Bの内容を，画面に表示しなさい，ということになります．最後のセミコロンは，画面に連続して表示させたいときに使用します．これに

より，行をかえずに次の表示を続けて出力します．

では，⏎キーを押して下さい．

〔8〕

187ページ参照

この行では，30行ですでに学んだ INPUT 文が再び使われています．ここでは文字によるキー入力をしてもらうようになっています．80行が実行されると，ディスプレイには「OK ？」のメッセージ文が表示されます．これにより，キー入力された文字（7字以内）を，文字変数C$に取り込みます．もしここで数値変数Cを使用したのなら，キー入力は数値しか受け付けず，このとき文字や⏎キーが入力されると SN エラーとなってしまいます．

この行の役割は， INPUT 文がキー入力があるまで待っている，という機能を利用して，70行の計算結果の表示をいったん止めてしまうことです．もしこの行がないと，プログラムの繰り返しによる多くの表示結果が一度に出てしまいます．

この行で，30行で入力した値の倍数を表示します．では，⏎キーを押して下さい．

〔9〕

182ページ参照

無条件に40行へ行け，という命令です．では，⏎キーを押して下さい．

〔10〕

```
100 END
```

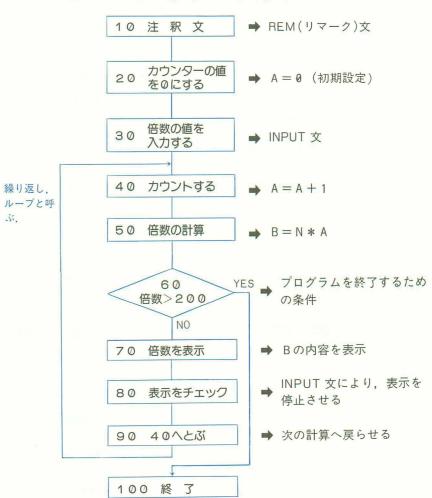
170ページ参照

END は，プログラムを終了せよ，という命令です．END は，プログラム上欠かせない命令なのです．というのは，このあとにさらにサブルーチンのプログラムが続くことが多いからなのです．この場合，END がないと，続けてこのサブルーチンを実行してしまうのです．サブルーチンについて

は，179ページを参照して下さい．

ENDは，次のように60行に入れることもできます．そうすれば100行は不要になります．では ⏎ キーを押して下さい．

```
60 IF B>200 THEN END
```

ここで，プログラムの流れをみてみましょう．

このようになります．ここで，90行は無条件に40行へ戻りますから，このプログラムは，40行～90行を繰り返し実行することになります．そしてこのループの途中に入れた，60行の条件文によって，このループから脱出し，100行へとぶのです．上記した流れ図を，フローチャートと呼びます．では，↵キーを押して下さい．

練　習　問　題

■ 数値を次つぎと入力していき，それを加算していくプログラムを作って下さい．

ヒント

① 変数をクリアして，0にしておく．

② 加算する値がいくつか聞く．

③ その数を，前の数にたす．

④ 結果を表示する．

答

```
10  CLEAR
20  INPUT  "DATA =";A
30  B = B + A
40  PRINT  "TOTAL =";B
50  GOTO  20
```

解説　10行の CLEAR はすべての数値変数と文字変数をクリアする命令です(B＝0として，積算用の変数Bのメモリーに0を代入するだけでもかまいません)．50行の GOTO 命令の行き先行番号を10とすると，そのつど，変数が0になり，積算ができなくなります．

プログラムの実行　3-8

さて，以上で入力したプログラムのチェックが終りました．説明の通りにキー操作していただきましたでしょうか．BASIC プログラムが，どのように作られているのか，だいたいのところを理解していただいたでしょうか．では，↵キーです．

それでは，このプログラムを実行させてみましょう．

まず，BRK キー，CLS キーを押して下さい．次に，プログラム実行命令 R U N ↵ もしくは，SHIFT RUN/Q ↵ を押して下さい．また，P0 のエリアにプログラムを入れたのなら，SHIFT P0/Ø でも RUNと同じ働きをします．

プログラムが正しく入力されていれば，次の表示が出ます．

```
RUN
BAISUU  WA  ?
```

倍数をいくつにするかを聞いてきます．次のキー操作をして下さい．

17　　ENTER

すると，次の表示が出ます．

```
RUN
BAISUU  WA ? 17
 17  OK ?
```

17の最小の倍数が，17であることを表示しています．また，確認はよいかを聞いてきます．それでは次の倍数を表示させましょう．↵キーを押して下さい．

```
RUN
BAISUU  WA ?  17
 17 OK ?
 34 OK ? _
```

さらに次の倍数は，↵キーを押します．以下この繰り返しで，187まで表示します．さらに，↵キーを押すと，突然次の倍数表示が出なくなってしまいました．これは，プログラムの60行の条件が成立し，倍数が200を越えたため実行が終了し，停止しているからなのです．再びこのプログラムを実行させるには，もう一度 SHIFT RUN/Q ↵ の操作が必要です．

〔プログラムの変更〕

ここで使用したプログラムに，さらにいろいろ手を加えて，各種の命令の使い方の理解を深めてみましょう．

まず，300までの倍数を出すようにしてみましょう．それには，60行のB＞200をB＞300に変更すればよいのです．

SHIFT EDIT/E 60 ↵

で，60行が出ます．カーソルを移動し，200を300に変更して↵キーを押し，BRK キーを押し，SHIFT RUN/Q ↵ です．

次に，70行の PRINT 文を変えて，表示の変化を調べてみましょう．

70行は，「PRINT B;」となっています．Bのあとのセミコロンは，表示を続けるという役割でした．では，このセミコロンをとってみましょう．

```
プログラムの実行順
BAISUU  WA?17
17  OK?
34  OK?
51  OK?
68  OK?
85  OK?
102  OK?
119  OK?
136  OK?
153  OK?
170  OK?
187  OK?
Ready  P0
```

SHIFT EDIT/E 70 ↵

で，70行を表示させ，カーソルを移動し，；の所で DEL キーを押します．";"が消えましたか．↵キーの後 BRK キーを押し実行させて下さい．RUN の方法は，もうおわかりと思います．

表示は，80行のメッセージ文「ＯＫ？」が，倍数の下に表示されます．つまり";"がなくなったので，次の表示を続けず，行がえして表示したことが分ります．

表示画面の作り方 3-9

それでは，入力してある倍数計算プログラムの70行を，さらにいろいろ変えて，画面表示のコントロール方法を学んでみましょう．

BRK SHIFT EDIT/E 70 ↵

で，70行の次の表示が出ましたか．

70 PRINT B

195ページ参照

では，プログラムの繰り返し回数のカウンター用変数Aも同時に表示させてみましょう．

〔1〕 70 PRINT A ; B ;

変更の方法は，カーソルをBの位置に移動させ，SHIFT INS/DEL INS/DEL です．これにより，Bの前に2字分スペースが空きますから，ここでA SHIFT ; ➡ SHIFT ; を入力します．次に ↵ キーを忘れないように．それでは，RUN させて表示を確認して下さい．

こんどは，次のように変更してみます．

〔2〕 70 PRINT A , B

変更の方法は，まず BRK 後 EDIT モードで70行を表示させ，カーソル移動で訂正し， ↵ の手順で上記と同様です．訂正がすんだら RUN させて下さい．「A，B」によって，表示が行改えとなりました．

さらに，次のように変更してみましょう．BRK キーを押して下さい．

〔3〕 70 PRINT A ; TAB (8) ; B ;

このように変更するには，上記と同様 EDIT モードにしカーソルを移動させ，訂正し， ↵ キーです．次に，RUN して下さい．

TAB（8）という命令が出てきました．これは，（ ）中に指定した桁までスペースを入れるという関数です．AとBの変数の内容表示の間にスペースがあいたのを確認して下さい．

さて，ここで困った問題が生じることにお気づきでしょうか．それは，数値変数Bの表示で3桁の数字になると，下のように左ぞろえとなり，最後が1字分ずれてしまうのです．

```
3              51  OK?
4              68  OK?
5              85  OK?
6              102  OK?
```

この場合は倍数の表示ですから，これでもよいかもしれませんが，例えば数量や価格を表示させる場合には，右ぞろえが見やすいでしょう．

そのためには，USING 関数を使用して，次のように書きかえます．

〔4〕　70　PRINT　A；TAB(8)；USING"＃＃＃"；B；

USING の使用法は229ページを参照して下さい．では，RUN して，表示を確認して下さい．

画面表示の制御を行なう命令は，他に LOCATE 命令があります．では，これを使用して，70行を同様に書きかえてみましょう．

〔5〕　70　CLS：LOCATE 5, 2：PRINT A；B；

（CLS 命令，LOCATE 5, 2 命令，PRINT A；B； 命令）

…このように，1つの行に2つ以上の命令を「:」によって接続させる行を，マルチステートメントといいます．

入力後，↵ キーを忘れないようにして下さい．表示は次の通りですか．

```
1  17  OK?
```

画面中央部に表示されます．

LOCATE 命令の使い方は，次のようになります．

LOCATE　X，Y

Xはよこの文字数，Yはたての行数の位置を書きます．詳しくは，194ページを参照して下さい．

指定した回数繰り返す　3-10

プログラムでは，ある決められた回数を，繰り返しある仕事（ルーチンといいます）を実行させる必要が頻繁に生じます．

例えば，たくさんのデータの中からある必要なデータのみを探し出す，あるいは，その多くのデータを数の多い順に並びかえるなどがそれに当たります．

こうした仕事を，コンピュータは，すべてのデータについて選択基準に合うか照合したり，となり合うデータを比べて大小を調べ，となり合うデータを入れかえて忠実に実行します．こんなとき，これから学ぶ命令が不可欠となるのです．

それでは，簡単なところからスタートしましょう．プログラムを入力する前に，まず，どのプログラムエリアが空いているか調べましょう．

SHIFT　SYSTEM S　↵　154ページ参照

この操作で，次の表示が現れます．

```
P      ♥♥23456789
 2056   Bytes :ANGLE 0
Ready  P0
_
```

注）RAMパックなしの標準実装で，プログラムを書き込まない状態のとき，2864Bytesを表示する．

♥のある所には，すでにプログラムが入っています．この表示の例では，P0とP1は使用中です．2056 Bytes というのは，残りのメモリー容量を表わしています．この場合，2056バイトをまだ自由に使用できるということです．残りのメモリー数は，増設RAMパックの数によって変動します．

ANGLEは，角度単位の設定（156ページ参照)で，0～2の値です．普通の計算をするときは，この数値が何であっても影響は受けません．電源ONしたときには，いつも0（DEGREE）に設定されています．

Ready P0は，現在のプログラムエリアを示し，P0にプログラムの書き込み，または実行が可能であることを示しています．では，空いているエリアにPROG数字で移行し，次のプログラムを入力して下さい．

```
10  CLS
20  FOR  A = 1  TO  20
30  PRINT  CHR$(135);
40  NEXT  A
```

正しく入力できましたか．確認のため，

と操作して下さい．間違いがなければ↵キーで次の行番号を表示させます．

Ready P0が出ましたら，プログラムをRUNさせて下さい．

このプログラムは，■を連続して20回表示させるものです．

〔1〕　　10　CLS

CLSは，画面の消去を行い，カーソルを左上の隅に移す命令で，いったん画面をきれいにしてから表示させたいときに使用します．

〔2〕

```
20  FOR  A = 1  TO  20
30  PRINT  CHR$(135);
40  NEXT  A
```

この20行と40行は，これでひとつの命令を構成しています．つまり，

FOR　A = 1　TO　20 ……変数Aに1から20までの数値を順番に代入する．

∫

NEXT　A ……A≦20であれば，FORに戻る．

これを，20～40行の FOR～NEXT ループと呼びます．実行手順を追ってみましょう．

① Aに1が代入される．

② 30行を実行する．

③ 40行の NEXT AでFORに戻る．

④ 20行へ戻り，Aに2を代入する．A＜20なので，次のNEXTまで実行し，再びFORへ戻れ，と命令する．

⑤ 30行を実行する．

⑥ 40行でFORへ戻る．

⑦ 20行へ戻り，Aに3を代入する．A＜20なので，次のNEXTまでを実行し，戻れ，と命令する．

⋮

こうして，Aに21が代入されると，A≦20が成立しないので，次のHEXTまで実行したら，もう戻らずに，次の命令へ実行を移す．この場合は，あとにプログラムがないので，Ready P数字となる．

さて，30行の20回繰り返される命令を見てみましょう．

〔3〕　30　PRINT　　CHR$(135);　　212ページ参照

表示しなさい　CHR$(135)を　↑— 続けて

CHR$ (135)は，キャラクターダラーの135番という意味で，使うことのできる文字やキー（キャラクター）に，すべて0～255までの番号が付けられています．(巻末表307ページ参照)．

「■」の表示は，直接にキー入力できませんから，CHR$（　）で指定して使うのです．このように，とくにキー入力できないキャラクターには，欠かせない命令ですから，(　)の中に32～254までの数字を入れて，その使い方に慣れて下さい．「CHR$(」は，SHIFT　CHR$(A　です．

練　習　問　題

■ FOR～NEXTを使用して，1から任意に指定した数までの整数を，連続して表示させるプログラムを作って下さい．

ヒント

FOR　A＝1　TO　N

```
〜
NEXT  A
```

答

```
10  CLS
20  PRINT  CHR$(182);CHR$(189);
    CHR$(222);
30  INPUT  N
40  FOR  A=1  TO  N
50  PRINT  A;
60  NEXT  A
70  END
```

解説 20行で，任意の数を入力してもらうため，INPUT の前にメッセージ文として「カズ？」の表示をさせるようにしました．これにより入力された数値を，数値変数 N に取り込んで，40行でFORの繰り返し数としました．Nには例えば15などの数値が入力されますから，その回数だけ50行を実行します．

50行は，FOR～NEXT で使用される変数を，そのまま表示させたのです．これによって，FOR～NEXT 命令で使用される変数に，ループの繰り返しごとに1が加えられていくのが分ります．

金額合計プログラムを作る　3-11

FOR～NEXT 命令は，慣れないとなかなか理解しにくいものです．そこで，もうひとつプログラムを作って，それに慣れることにしたいと思います．

まず，次のプログラムを入力するプログラムエリアを決めましょう．ここでは仮にＰ４とします．入力手順は，もう改めて記す必要のないものと思います．では，NEW ↵ 後，次のプログラムを入力して下さい．

このプログラムは，最初に決めた件数を入力すると，あとは単価，数量を次つぎと件数分繰り返し入力していくだけで，小計と合計を表示してくれるものです．

```
 10 CLEAR
 20 INPUT "KENSUU";N
 30 FOR A=1 TO N
 40 INPUT "TANKA";B
 50 INPUT "SUURYO";C
 60 PRINT "SHOUKEI";B*C
 70 D=D+B*C
 80 NEXT A
 90 PRINT "TOTAL";TAB(10);CHR$(92);D
100 END
```

このプログラムは，各行番号で，次のように仕事を処理しています．

10　すべての変数をクリアつまり０を代入します．

20　処理する件数を入力します．(件数をＮに入れる)

30～80　FOR～NEXT ループで，件数(Ｎ)回，単価，数量を表示させ，入力し，小計額を表示し，合計額に加算する計算を行ないます．

90　金額合計の表示

100　終了

ここで，30行から80行の FOR～NEXT ループを，詳しく考えてみましょう．FORからNEXT までのループの中でやらせたい仕事は，

① 単価を入力して，変数Bに入れる．

② 数量を入力して，変数Cに入れる．

③ 単価×数量の計算を行ない，小計を表示させる．

④ 合計額に，小計額を加算し，新しい合計額とする．

………ことなります．

90行のプログラムは，表示の見やすさに配慮してあります．

```
90  PRINT "TOTAL" ; TAB(10) ; CHR$(92) ; D
```

TOTALを表示させ，　10個スペースをとり，　¥を表示させ，　合計額を表示する．

それでは次に，FOR～NEXT ループの中の仕事は，ひとかたまりのまとまった仕事なので，プログラム上これを分離独立させ，その仕事のところへプログラムが進んだら，そのブロックへ飛んで仕事をこなし，終了したら元の所へ戻る，というサブルーチンの考え方をとりいれて，プログラムを作りなおしてみることにしましょう．同時に表示のみやすさにも工夫をしてみます．

このサブルーチンの考え方は，右の図のように考えます．このために使用する命令は，GOSUB～RETURNです．

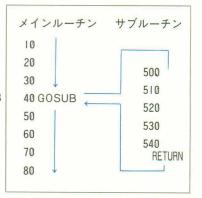
前のプログラムの40～70行を，右図に従って500～540行に変更して40行にGOSUB 500を入れます．

```
GOSUB 500
```

500行のサブルーチンへ行きなさい．

```
RETURN
```

このサブルーチンを呼び出したGOSUBの次の命令へ戻りなさい．

これによって，30～80行は，次のように変更します．

```
30 FOR A=1 TO N
40 GOSUB 500
50 NEXT A
```

それでは，これらをまとめてひとつのプログラムにしてみます．

```
10 CLEAR
20 INPUT "KENSU";N
30 FOR A=1 TO N
40 GOSUB 500
50 NEXT A
60 PRINT "TOTAL";TAB(10);CHR$(92);D
70 END
500 PRINT A;TAB(5);"TANKA";:INPUT B
510 PRINT A;TAB(5);"SUURYO";:INPUT C
520 PRINT "SHOKEI";B*C:BEEP 1
530 D=D+B*C
540 RETURN
```

FOR／NEXT のループ．N回繰り返す．（30～50行）

サブルーチン（500～530行）

このプログラムは，前のプログラムを EDIT モードで手直しして入力するよりも，変更が多いので，新しいプログラムエリアに入れてみて下さい．そして，前のプログラムと，実行上の相違を確認して下さい．

文字変数の練習　3-12

これから，「指定した回数繰り返す」の項の頭で述べたデータをたくさんたくわえる方法について学んでいきますが，その前に文字のデータの扱いに慣れておく必要がありますから，少し文字変数の練習をしておきましょう．

変数には大きく分けて，次の2つがあることは，もうご存知ですね．

数値のみを代入することができる………**数値変数**(A, B, C, A1 など)

文字や記号を代入することができる……**文字変数**(A$, B$, A1$ など)

プログラム中での数値変数と文字変数のちがいは，何でしょうか．文字変数でいう「文字」とは，A, B, C………のアルファベットやカナや記号，そして0～9までの数字も含まれます．そうすると，数値と文字のちがいがますますあやふやになりそうです．

でも，この2つの決定的なちがいは，数値は大きさを表わしている，ということなのです．ですから，文字変数で扱われた場合の数値は，記号と同じで，大きさを表わすものではないのです．

では，このちがいを具体的に調べてみます．

〔数値の場合〕

4＋3 ENTER ➡ 7

〔文字の場合〕

"4"＋"3" ENTER ➡ 43

ここで，文字を扱う場合，必ず" "で囲むのは，INPUTやPRINT命令のメッセージ文などと同様です．この例でいえば，4という文字と3という文字をつなげて表示する，という意味になるのです．文字変数として扱っている数字を，数値として扱いたいというときには，VAL関数を使用して数値に変換することが可能です(214ページ参照)．

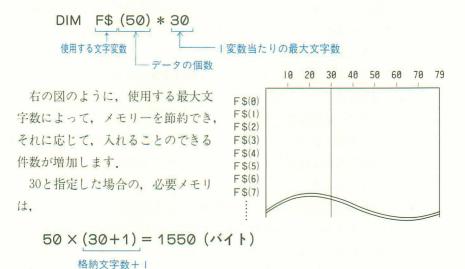
次項で述べる配列変数では，こうした文字変数をたくさん使用します．文字変数を配列変数で使用した場合，79文字までをひとつの変数に入れることができます(28ページ参照)．

3-12

文字配列の宣言は，次のようにします．

右の図のように，使用する最大文字数によって，メモリーを節約でき，それに応じて，入れることのできる件数が増加します．

30と指定した場合の，必要メモリは，

50 × (30+1) = 1550 (バイト)

となります．

配列プログラムでは，このように少しでも多くのデータを入力できるようにするために，1変数当たりの使用メモリー数をできるだけ少なくするように工夫します．

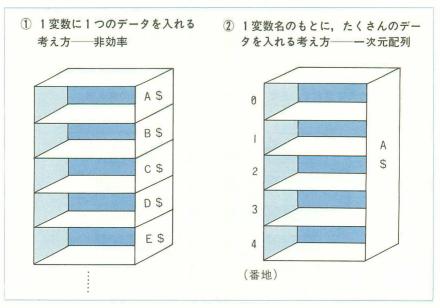
例えば，数値変数の場合は，このために半精度数値配列と単精度数値配列を選択でき，半精度数値配列であれば，単精度数値配列の半分のメモリーですみます．これについては次の項で詳しくお話しましょう．

ディメンションって何だろう　3-13

プログラムに携わる人たちの間で,「ディメンションをとる」と話しているのを聞いたことはありませんか．これは「入れ物を用意する」という意味です．その入れ物というのは，右下の②のような整理棚を思い浮べて下さい．

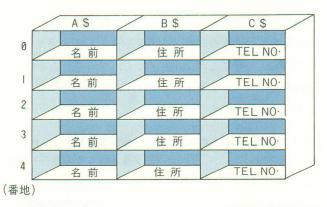
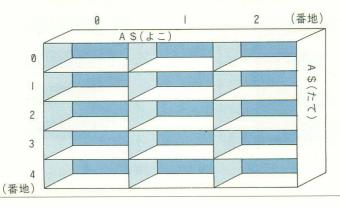
それぞれの棚には，データ（保存しておきたい数値や文字列）を入れることができます．

ひとつの変数名のもとに，この例では０～４の５つの棚が用意されることになります．この整理棚の使用法は，例えば文字変数Ａ＄の０番には「TANI」，Ａ＄の１番には「SUZUKI」，２番には「YAMAMOTO」………というように，いろいろな名前をひとつの変数の標識のもとに管理することになりますから，必要な名前をこの管理番号によって呼び出すことができるのです．

もし，こうしたひとつの変数名に番号を振って，データを蓄えることをしないと，①のようにひとつの変数にひとつのデータを入れる,ということになり，変数名がいくつあっても足りないばかりでなく，プログラムを作る上でも，どのデータがどの変数に入っているか管理できず，非効率きわまりないものになってしまいます．

■ 一次元配列

そこで，データをたくさん入れるには②の方法をとるのですが，これを**一次元配列**と呼びます．つまり，ひとつの変数名のもとに，一方向に番号を振って，データを配列させる，という意味です．図では0～4までですが，実際には，全部で0番地から255番地の256段の棚をつくることができます．

こうしたデータの配列を使用するときには，例えば，A$ の変数名のもとに整理棚が最大いくつ必要なのか，メモリーを予約しておく必要があるのです．この予約をしておかないと，コンピュータは，データが入ってきたときに，ど

③ 一次元配列をいくつか並べた住所録の場合.

④ ひとつの変数名で，たて・よこを表わす二次元配列の考え方.

このスペース(メモリー)にそのデータを入れておくべきか分らず，UVエラーとなってしまいます．要するに，A$ という名の整理棚を何段作って用意しておくべきか分らない，ということになるのです．

A$ という名の整理棚の長さ，容積を，A$ のディメンション（寸法，大きさ）といいます．

■ 二次元配列

さて，それでは，整理棚の形を少し変えてみましょう．

一次元配列では，③のように例えば A$ の配列棚には名前， B$ の配列棚には住所， C$ の配列棚には電話番号を入れると決めれば，名簿や住所録にすることができます．プログラム上は，名前を入力すると，その名前の番地の B$ と C$ を呼んでくればよいことになります．

こうした整理棚では各棚の管理の方法として，よことたてに番地をつけ，よこ何番，たて何番と指定すれば，それがどの位置の棚であるかをいうことができるようになります．つまり，よこ番地とたて番地の組み合わせで棚位置を指定する方法です．

この考え方を図で表わすと，④のようになり，この表わし方を**二次元配列**と呼ぶのです．ひとつの変数名で，たて，よこという二方向に番号を振って，データを配列させる，という意味です．

この場合，たては 0 番地から 255 番地まで，よこも 0 番地から 255 番地までとることができます．しかし，たて・よこともに 255番地までとったりすることはできません．それは，メモリー容量に限度があるからです．

二次元配列でのメモリーの使用量は，

たて番地数×よこ番地数×1番地当たりの必要メモリー数

です．このため文字配列に入力できる文字数を，あらかじめ設定しない場合には，1変数あたり16文字となりますから，

256 × 256 × 17 = 1114112 バイト

となり，完全にメモリーオーバーとなってしまいます．

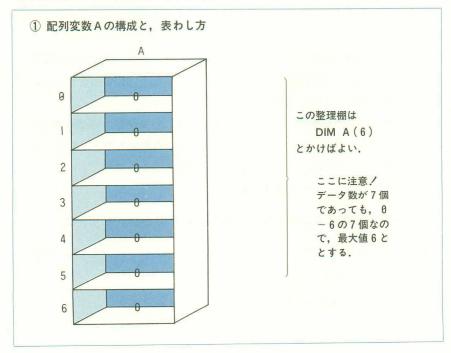
数値配列変数の練習 3-14

数値配列変数とは，前項で考えた配列という整理棚に，数値を入れる場合の変数の使い方を指しています．ひとつの変数の名で，たくさんのデータを整理する変数を，配列変数といいます．前項の整理棚のたとえでいえば，A$, B$ C$ などが配列のための変数――配列変数に当たります．

ところで，数値を管理するための配列変数と文字列を管理するための配列変数の使い方は，基本的にはまったく同じですが，プログラムの中では，データの扱い方や表わし方がいろいろ異なるのです．

そこで，ここではまず数値配列変数のプログラム作りから学んでいきます．

①のような一次元配列の整理棚をもう一度考えましょう．この整理棚に名前をつけましょう．この名前は，アルファベット1文字，つまり26個の固定変数でなければなりません．ここでは，仮にAとします．

そうすると，この場合，Aという配列変数の整理棚は，0～6までの7つの棚で構成されていることになります．

そこで，コンピュータにあらかじめ "Aという名の，0～6の棚をもった整理棚を作りなさい" つまり，Aを配列変数として使用し，その大きさ（ディメンション）は0～6である，ということを宣言しておかなくてはならないのです．

その方法を，次に示します．

このように指定すると，数値配列変数Aのすべての配列が0になります．

それでは，上の操作を行なって下さい．もしここで，DD エラーが出た場合は，一度 CLEAR ↵ してから，上の操作を行なって下さい．

次に，①の各棚にいろいろなデータを入れてみましょう．

A (5) ＝ － 13 ↵

（A (5)：配列変数Aの5番地）

A (3) ＝ 65 ↵

次に，A(5) と A(3) の棚に，前記操作でデータを入れることができたかどうか確認してみましょう．

A (5) ENTER ➡ －13と表示

A (3) ENTER ➡ 65と表示

以上までの操作を，プログラムに書くと，次のようになります．

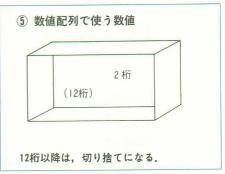
```
 5  CLS
10  CLEAR
20  DIM  A(6)
30  INPUT  "A(5) = "; A(5)
40  INPUT  "A(3) = "; A(3)
50  PRINT  A(5); A(3)
60  END
```

■ 配列変数で使用する数値の知識

以上で，数値のデータを配列の整理棚に格納する方法は分りました．では，配列変数に入れることのできる数値は，どんな制約を受けるのでしょうか．まず大切なことは，ひとつの配列変数には，12桁の整数部と２桁の指数部を入れることができるということです．

ところが，画面表示をすると，11桁目を四捨五入した10桁表示となってしまうことに注意して下さい．計算は12桁で，表示は10桁までです．

⑤ 数値配列で使う数値

12桁以降は，切り捨てになる．

こうした数値配列変数で使用される数値を，**単精度**と呼びます．これに対して，有効な数字が12桁も必要ないし，表示も10桁を必要としない，５桁あれば十分である，ということも少なからずあるでしょう．こんなときに，大変便利な数値の格納法がありますので，是非使いこなして下さい．数値を最大５桁で配列変数に格納する**半精度**と呼ぶ数値の処理法です．これによれば，１データ

桁数の表示

PB―700は，数値は100桁まで表わすことができます．また，プログラムの力によって，もっと多くの桁表示も不可能ではありません．ただし，10桁を超えると，画面に表示されたときに，指数表示となります．

例えば，

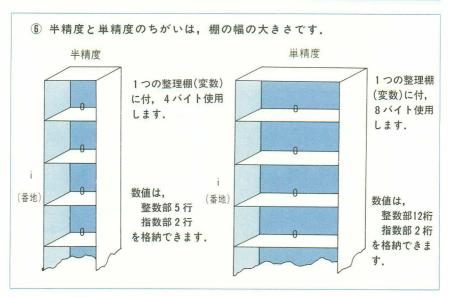
12345678909 ENTER ➡ 1.234567891 E 10

整数部　四捨五入します．　指数部

指数表示とは，$\times 10^{10}$ の表現のことです．上の例では，

1.1234567891 × 10^{10}

を意味します．

⑥ 半精度と単精度のちがいは，棚の幅の大きさです．

当たりの必要なメモリーが，単精度の半分（4バイト）ですむのです．

単精度と半精度のちがいをまとめると，⑥のようになります．

単精度のディメンションの表わし方は，すでに述べたように変数がAの場合，

A (i)　　…………　単精度数値配列の宣言

ですが，半精度のディメンションの表わし方は，次のようにします．

A ! (i)　　…………　半精度数値配列の宣言

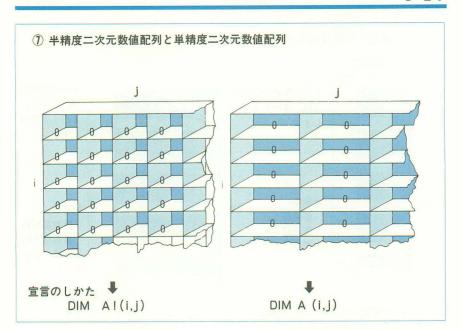
変数の後に！が必要です．

■ 二次元配列における半精度と単精度の考え方

考え方は，前記一元配列におけるのとまったく同様です．整理棚に例えると，⑦のようになります．

宣言のしかたは，数値配列変数Aの場合，

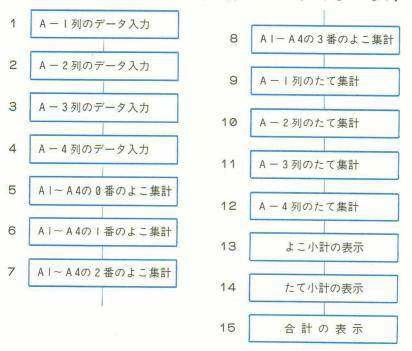
DIM A ! (i, j)　……　半精度　　　DIM A (i, j)　……　単精度

⑦ 半精度二次元数値配列と単精度二次元数値配列
j
i
0
j
i
0
宣言のしかた
DIM A!(i,j)
DIM A (i,j)

数値配列プログラミング入門　3-15

下の表のようなたて・よこ集計表のプログラムを作ってみましょう.

たて，よこの小計を計算し，最後に合計を得るものとします．たて，よこ各4個，計16個のデータとしましょう.

	A－1	A－2	A－3	A－4	小 計
0	25	17	3	67	
1	19	20	11	58	
2	32	15	26	55	
3	36	28	29	40	
小計					

データは数値ですから，まず，一次元数値配列でプログラムを作りましょう．変数名はAでやってみることにします．プログラムを作る手順は，次のようにしましょう．こうした手順を書いたものを，フローチャートといいます.

ここで，フローチャートの1～4番の各列のデータ入力プログラムを考えてみましょう．まず最初の一列は，

```
10  CLEAR
20  DIM  A!(15)
30  FOR  I=0  TO  3
40  INPUT  A!(I)
50  NEXT  I
```

（30～50：FOR～NEXT ループ）

このプログラムで，A－1列の0～3番にはデータを入れていくことができますが，さてA－2列はどうしたらよいでしょうか．ここでは別の変数を使用せず，Aのみを配列で入力していく方法を考えます．

→J					小 計
↓I	A!(0)	A!(4)	A!(8)	A!(12)	B!(0)
	A!(1)	A!(5)	A!(9)	A!(13)	B!(1)
	A!(2)	A!(6)	A!(10)	A!(14)	B!(2)
	A!(3)	A!(7)	A!(11)	A!(15)	B!(3)
小 計	C!(0)	(!)(1)	C!(2)	C!(3)	D

このようにA!(0)～A!(15)の棚に順番にデータを入れていき，最後に各集計を行なうには，どんな考え方のプログラムであればよいか，これをプログラムのアルゴリズムと呼びます．ここで，いっそのことA!(0)～A!(15)を，ひとつの長い棚のように扱い，4個おきのデータをたしていけば，よこの小計を得ることができる点に着目した方法を次に示します．

```
10  REM  NYURYOKU
20  CLEAR
30  DIM  A!(15)
40  FOR  J=0  TO  3
50  FOR  I=0  TO  3
60  PRINT  "A-";J+1;"(";I;")";
70  INPUT  A!(J*4+I)
80  NEXT  I
90  NEXT  J
```

（50～80：たてにデータを入力）
（40～90：A－1列から順に列をずらしてデータを入力）

これで，データの入力プログラムができました．「A－列番号(たて番号)？」の表示で，順にデータ入力していきます．70行の配列変数 A!( )に，データを格納していくのです．

次は，よこの小計用のプログラムを作りましょう．

```
 95  DIM B!(3)
100  FOR  I=0  TO 3
110  FOR  J=0  TO 3
120  B!(I)=B!(I)+A!(J*4+I)
130  NEXT J
140  PRINT "B-";I;TAB(5);B!(I)
150  INPUT "OK";F$
160  NEXT  I
```

（110～150行：よこの計算と表示　100～160行：たてに4回ずらして，小計を計算，表示）

配列変数 B!(0)～B!(3)に，120行でデータを格納していき，140行でそれを表示させます． 150行は，B!(0)～B!(3) の小計の表示が，スクロールされてしまうのを防ぐため，表示を止め，ENTERキー入力で次の表示がでるようにしてあります．

次は，たての小計を行なうプログラムです．

```
170  DIM C!(3)
180  FOR  J=0  TO 3
190  FOR  I=0  TO 3
200  C!(J)=C!(J)+A!(J*4+I)
210  NEXT  I
220  PRINT "C-";J;TAB(5);C!(J)
230  INPUT "OK";F$
240  NEXT J
```

（190～230行：たての計算と表示　180～240行：よこに4回ずらして，小計を計算，表示）

よこ小計の方法とほとんど同じで，ダブルの FOR～NEXT ループによっています．異なるのは，I とJ の変化（ループ制御変数といいます）の順番だけです．

それでは，合計を計算してみましょう．

```
250  REM   GOUKEI
260  FOR   I =0 TO 3
270  D = D+C!(I)
280  NEXT   I
290  PRINT "TOTAL";D
300  END
```

二次元配列による方法

一次元配列を使用すると，いままてきたようにデータを格納するべき各棚の扱いが複雑で，わかりにくくなります．そこで，すでに学んだ二次元配列によって，プログラムを作ってみましょう．こうした，たて・よこのデータを扱うには，二次元配列は大変便利なのです．

1 初期設定

```
10 ERASE A!         ……変数Aの配列をクリアする．
20 CLS              ……画面をクリアする．
30 N=4              ……たて，またはよこの数
40 DIM A!(N,N)      ……ディメンションの設定(半精度数値配列設定)
```

30行のN＝4は，この数字を変えれば，たて・よこの数を変更できます．

2 データの入力

```
50 FOR I=0 TO N-1
60 FOR J=0 TO N-1
70 PRINT I;"-";J;
80 INPUT A!(I,J)
90 NEXT J
100 NEXT I
```

I↓ / →J					よこ計
	28	39	12	54	133
	53	29	55	30	167
	28	17	80	53	178
	60	31	70	44	205
たて計	169	116	217	181	683

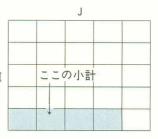
50行と60行の FOR~NEXT 文のN－1は，データ入力では合計欄が必要ないので，－1したのです．データ入力の順番は，前の一次元配列とは変えて，こんどは横へ順番に入力していくことにしました．

3 よこの小計

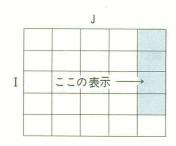
```
110 FOR I=0 TO N-1
120 FOR J=0 TO N-1
130 A!(I,N)=A!(I,N)+A!(I,J)
140 NEXT J
150 NEXT I
```

よこの小計を4回(N＝4のとき)行ないます．IとJに0～N－1の数を入れて，A！(　)の値を確認してみて下さい．

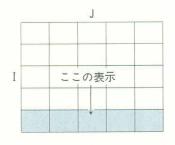
4 たての小計

```
160 FOR J=0 TO N
170 FOR I=0 TO N-1
180 A!(N,J)=A!(N,J)+A!(I,J)
190 NEXT I
200 NEXT J
```

たての小計を4回行ないます．IとJに0～N－1の数を入れて，A！(　)の値を確認して下さい．

5 よこの小計の表示

```
210 FOR I=0 TO N-1
220 PRINT A!(I,N);
230 NEXT I
240 STOP
```

いったんストップをかけ，よこの各小計を確認した後，SHIFT CONT W ↵ で，次の行へ実行が移ります．

6 たて小計の表示

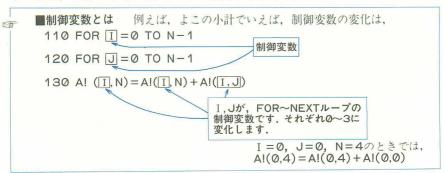
```
250 FOR J=0 TO N
260 PRINT A!(N,J);
270 NEXT J
280 END
```

たて小計の表示を行ないます．このときよこ小計の合計を同時に行ないますから，全部の合計を同時に表示することになります．

さて，たて・よこ集計プログラムでむずかしいのは，FOR～NEXT ループを二重に使うことです．しかしこれも，FOR～NEXT 命令を何度か自分で試してみて慣れるに従い，大変便利でプログラムの簡素化に役立つものであることが分かってくるでしょう． FOR～NEXT ループで大切なことは，制御変数に自分で0から順番に数値を入れていってみることです．

7 プログラムの実行

RUNすると，「0－0？_」と聞いてきます．データ ⏎ キーで「0－1？_」と次の横の欄へ入れる値を聞いてきます．全部のデータの入力を終了すると，しばらくしてよこの小計と，「STOP P0－240」の表示が出てストップします．

次に，SHIFT CONT W ⏎ で，たての小計と合計を表示して終了します．表示の形は，自分で手を入れていろいろ試して下さい．

文字配列変数の練習　3-16

文字を入れることのできる文字配列変数においても，数値配列変数のときと同様に，一次元配列と二次元配列を使用することができます．数値配列変数では，さらに半精度と単精度を使用することができ，メモリーの有効使用に役立ちますが，文字配列変数においても同じことがいえます．

5文字ぐらいでよい配列変数に，6文字以上も入れることのできるスペースを取ってもムダなことですし，反対に20文字入れたいのに，そのスペースが10文字分しか取っていなければ，データ入力のとき困ってしまいます．つまり，文字配列では，格納できる文字を1～79まで指定できるという特長をうまく使いこなすことが大切になります．

ひとつの文字配列変数当たりの文字数の指定は，次のように行ないます．

DIM A$(i)＊n　(1≦n<79)
※＊nを省略すると16文字に設定されます

■DIM文でのエラー

DIM文で指定した以上の文字を入力しようとすると，

ST－error

が出ます．このときは，限度内の文字数を再入力して下さい．
また，DD-error が出ることもあります．これは，以前に(スイッチを切っても関係なしに)，これから指定しようとする配列と同じ変数名で，指定が行なわれていた場合に生じます．例えば以前に，

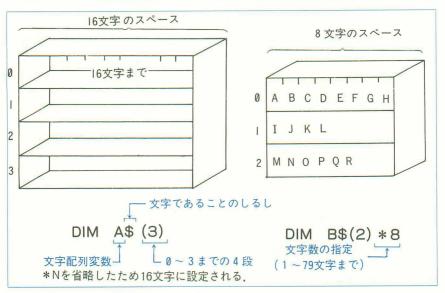
DIM A$(2Ø) ↵

をどこかで実行していた場合，新たに，

DIM A$(3Ø) ↵

を実行しようとすると生じるのです．このエラーが出たときは，ERASE 命令でA$ を消去して下さい．例えば，

ERASE A$ ↵

ただし，ERASE や CLEAR を実行すると，データが消えてしまいますから注意が必要です．

それでは，さっそく配列変数に文字を入れてみることにしましょう．次の操作を行なって下さい．

```
CLEAR ⏎
DIM  H$(30)*5 ⏎
H$(1) = "ABCD" ⏎
H$(2) = "EF" ⏎
```

以上を実行すると，右の図のように，文字が配列変数に格納されます．

これを呼び出すには，次の操作を行ないます．

■文字を配列させるプログラム

練　習　問　題

問　題　最大10文字を収容できる，配列数が3つの配列プログラムを作りなさい．なお，文字はキャラクタ・コードで入力し，途中で入力をやめたいときは，0を入力，3つの配列に入力終了すると，すべてのキャラクタを表示させるものとします．

ヒント　初期設定は，最大10文字，配列数が3ですから，

```
10  CLEAR
20  DIM  N$(2)*10
```

次に，データの入力用ルーチンです．配列数が3なので，3回のFOR～NEXTループを作りループの中で，文字を10個読むためのGOTO文のもうひとつのループを回せばよいことになります．その中にINPUT文を置き，キャラクタ・コードを読むのです．

データ入力プログラムの基本構成は，次のようになります．

①1変数当たりの文字数のカウンターを0とする　…… B=0
②キャラクタ・コードを入力　…… INPUT N
③文字数のカウンターに1をプラス　…… B=B+1
④カウンターが10を越えたら，次の配列変数へ入力…… IF B=10 THEN～
⑤配列変数に，コードNo.を文字に変換して格納　…… N$(I)=N$(I)+CHR$(N)
⑥②へ戻る　…… GOTO～

次は，結果の表示のルーチンです．N$(0)～N$(2)を続けて表示するだけです．

```
100  FOR  I=0  TO  2
110  PRINT  N$(I);"␣";……… Iに0～2が順次入る.
120  NEXT  I
```

答

```
10 CLEAR :CLS
20 DIM N$(2)*10
30 FOR I=0 TO 2:B=0
40 PRINT "N$(";I;")";:INPUT "  No.";N
45 IF N>255 THEN 40
50 IF N=0 THEN 90
60 B=B+1:IF B=10 THEN BEEP :GOTO 90
70 N$(I)=N$(I)+CHR$(N)
80 GOTO 40
90 NEXT I
100 FOR I=0 TO 2
110 PRINT N$(I);" ";
120 NEXT I
130 END
```

RUNさせると,「N$(0)　　　　NO. ?_」と表示されます．このときキャラクタ・コードを入れていくのです．文節で区切るときは，0を入れます．これで,「N$(I)」の中の数字が0～2へ変化していきます．

では，次のコードを入れていって下さい．サテ，何が出るでしょうか．

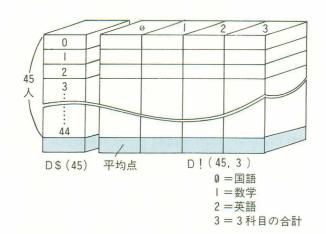
207 ↵ 194 ↵ 192 ↵ 222 ↵ 165 ↵ 190 ↵ 178 ↵
186 ↵ 0 ↵ 207 ↵ 175 ↵ 193 ↵ 0 ↵
183 ↵ 174 ↵ 221 ↵ 183 ↵ 174 ↵ 221 ↵ 0 ↵

文字配列と数値配列の組み合わせ　3-17

実際にたて・よこの表を作って，データの処理を行なうとき，文字配列と数値配列は，組み合わせて使用されることがほとんどです．このとき，文字の部分と数値の部分がひとつのデータ群として同時に取り扱わねばなりません．

例えば，氏名と得点，品名と個数や金額などを考えてみると，文字と数値をいっしょに呼び出せなくてはなりません．こうしたプログラムの方法を学んでみましょう．

わかりやすい例題として，成績処理プログラムを考えてみます．まず，氏名が必要です．次に，氏名に対応した国語，数学，英語の三教科の合計得点を記憶しておく二次元数値配列が必要です．

これから，次のような模式図を想定してみましょう．

入力したいデータは，次のものとします．

氏　名	国	数	英	得点計
A，Y	50	60	75	ア
K，K	83	71	70	イ
S，O	60	63	40	ウ
平均点	エ	オ	カ	キ

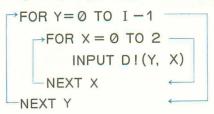
1 名前の入力

```
10 ERASE D$,D!:CLS
20 DIM D$(45)*10,D!(45,3)
30 I=0
40 INPUT "NAME ";D$(I)
50 IF D$(I)="END" THEN 80
60 I=I+1:IF I=45 THEN 80
70 GOTO 40
```

名前をD＄(０)～D＄(44)とし，平均点というタイトルをD＄(45)としました．「END」の入力でループからの脱出です．もういうまでもないと思いますが，40行～70行が，名前を入力するループです．

2 数値の入力

二次元配列によって数値データを入力していくルーチンです．まず，90行で名前（D＄(Y)）を表示させ，120行でその名前の国語（Xが０のとき）の得点を，D！(０，０）に取り込みます．その後，130行でその名前の得点をD！（０，３）に加算します．D！(Y，３)は小計欄ですが，もっとよこの科目を増やしたいときには，３を，その数に１をたした数に変えればよいのです．

D！(Y，X)のY，Xは，FOR～NEXTループの回数によって決定されます．

基本的には，このようなしくみになっています．

```
 80 FOR Y=0 TO I-1
 90 PRINT D$(Y)
100 FOR X=0 TO 2
110 PRINT X
120 INPUT " ";D!(Y,X)
130 D!(Y,3)=D!(Y,3)+D!(Y,X)
140 NEXT X
150 NEXT Y
```

→ X　　得点表

↓ Y

国(0)	数(1)	英(2)	計

3 課目平均の計算

氏名の入力回数（カウンターのＩ）がいくつであっても，Ｄ！(45，Ｘ）を課目の平均値欄としましたので，ここに，課目ごとの平均値を入れていくようにしましょう．まず，180行で，課目ごとの得点合計をＤ！（45，Ｘ）に代入し，200行で，平均値をＤに代入し，210行で，Ｄの小数点以下２桁目を切上げ，さらに，小数点以下１桁目以後を切捨てて，整数化し，その値を再びＤ！(45，Ｘ）に代入します．つまり，平均点の小数点以下１桁目が９であるときのみ切上げ，他は切捨ての処理をしています．INTは整数化の関数で，小数点以下切捨てとなります．

これは，成績処理という仕事の特質のためで，他の目的であれば，210行を変えることになります．

```
160 FOR X=0 TO 3
170 FOR Y=0 TO I-1
180 D!(45,X)=D!(45,X)+D!(Y,X)
190 NEXT Y
200 D=D!(45,X)/I
210 D!(45,X)=INT(D*10+0.5)/10
220 NEXT X
```

4 氏名別合計得点の表示

各氏名ごとの合計得点を表示して行きます．260行で，⏎キーが押されたら，次の氏名へ移るようにしました．氏名に対応した合計得点は，ループ制御変数Ｙで処理してあります．ですから，

氏名………… D$(Y)
合計得点…… D!(Y,3)

が，いつも一致していることになるのです．

```
230 FOR Y=0 TO I-1
240 PRINT D$(Y);" T=";
250 PRINT D!(Y,3)
260 K$=INKEY$:IF K$="" THEN 260
270 NEXT Y
```

5 平均点の表示

課目別平均点の表示は，D！(45, X)で，Xが0～3のデータを順次表示すればよいことになります．

```
280 FOR X=0 TO 3
290 PRINT "AVE=";
300 PRINT D!(45,X)
310 K$=INKEY$:IF K$="" THEN 310
320 NEXT X
330 END
```

では，RUNして，データを入力してみて下さい．実行結果の画面表示は，次のようになります．

```
A, Y T=  185 ┐
K, K T=  224 ├ ⏎キーを押していくと，順次氏名別得点計が表示されます．
S, O T=  163 ┘
AVE=  64.3 ——国語(0)の平均点 ⏎
AVE=  64.7 ——数字(1)の平均点 ⏎
AVE=  61.7 ——英語(2)の平均点 ⏎
AVE=  190.7——氏名別得点計の平均点 ⏎
```

カナ文字を使う　3-18

キャラクタ・コード表の通り，カナ文字や一部の漢字まで使用することができます．カナでいろいろなメッセージなどを表示すれば，グンと分りやすくなりますから，少々手間はかかりますが面倒がらずに大いに使うようにしましょう．カナ文字のみをアレンジしたコード表を次に示します．

カナ文字のコード表

ア	カ	サ	タ	ナ	ハ	マ
177	182	187	192	197	202	207
イ	キ	シ	チ	ニ	ヒ	ミ
178	183	188	193	198	203	208
ウ	ク	ス	ツ	ヌ	フ	ム
179	184	189	194	199	204	209
エ	ケ	セ	テ	ネ	ヘ	メ
180	185	190	195	200	205	210
オ	コ	ソ	ト	ノ	ホ	モ
181	186	191	196	201	206	211
ヤ	ラ	ワ	ァ	ャ	。	゛
212	215	220	167	172	161	222
	リ	ン	ィ	ュ	「	゜
	216	221	168	173	162	223
ユ	ル	ヲ	ゥ	ョ	」	
213	217	166	169	174	163	
	レ		ェ	ッ	、	
	218		170	175	164	
ヨ	ロ		ォ		・	
214	219		171		165	

上のコード表をみると，「ア」のコードは 177 です．従って，「ア」と書くのは，CHR ＄関数を使い，（　）でコード番号を囲んで，

```
PRINT CHR $ (177)
```

212ページ参照

でよいのです．続けて「カズ」と書くには，

PRINT　CHR$ (182) ; CHR$ (189) ; CHR$ (222) ↵

ですし，「ペン」と書くには，

PRINT　CHR$ (205) ; CHR$ (223) ; CHR$ (221) ↵

です．「CHR$(」のワンキー入力を利用するのは，いうまでもありません．ちょっと分かりにくいコードを次にあげておきますので，キー入力して確認して下さい．

〈コード番号〉	〈意　味〉	〈表示〉	〈コード番号〉	〈意　味〉	〈表示〉
160	空白		165	中点	・
161	読点	。	176	長音	ー
162	カギカッコ	「	222	濁点	ﾞ
163	カギカッコ	」	223	半濁点	ﾟ
164	句点	、			

これらの記号も，すべて一文字の扱いとなります．また，ワ行の「ヲ」は，166であり，167から175までは「チョット」などというときの小さいカナ文字になっています．

■ **長い文章を表示させるためのプログラミング法**

長い文章を入れたいとき，CHR $(　)を続けていくと，やたらプログラムが長くなってしまいます．そこで，DATA文にコードを書いて，READ命令で読み込むようにするのが簡単です（200ページ参照）．例えば，「ウリアゲダカシュウケイ　ヒョウ」と書くのは，次のプログラムの40行のGOTO命令によるループで，1字ずつDATA文から変数Aに読み込み，30行でそれを表示させます．0を読み込むと，終了します．

```
10 READ A
20 IF A=0 THEN END
30 PRINT CHR$(A);
40 GOTO 10
```

```
50 DATA 179,216,177,185,222,192,222,1
   82,160,188,173,179,185,178,160,203
   ,174,179
60 DATA 0
```

それでは，もうひとつカナ文字を使用した事例をあげておきます．

```
10 CLS
20 PRINT CHR$(203);CHR$(221);CHR$(210
   );CHR$(178);:INPUT " ",N$
30 PRINT CHR$(189);CHR$(179);CHR$(216
   );CHR$(174);CHR$(179);:INPUT " ",Q
40 PRINT CHR$(192);CHR$(221);CHR$(182
   );:INPUT " ",U
50 PRINT CHR$(183);CHR$(221);CHR$(182
   );CHR$(222);CHR$(184);CHR$(32);
60 PRINT CHR$(202);Q*U;CHR$(241);CHR$
   (32);CHR$(195);CHR$(222);CHR$(189);
70 IF INKEY$="" THEN 70
80 END
```

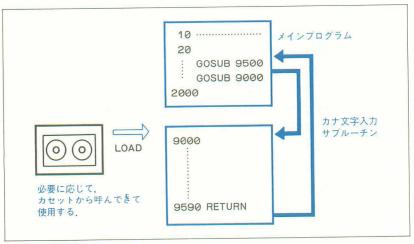
実行例

```
ヒンメイ    PB-300
スウリョウ  20
タンカ      29800
キンガク ハ 596000円 デス
```

カナ文字を直接入力する方法

カナ文字をキャラクタ・コードで入力していくのは，煩しいと感じる読者に，大変便利な方法があります．それは，プログラムによって，アルファベットキーやテンキーの入力をカナに変換して表示し，それを配列変数にたくわえる方法です．

そのプログラムを紹介しましょう．

まず，このプログラムは，行番号9000以後となっています．ですから，キー入力したら，カセットへセーブしておき，必要に応じてメインプログラムに重ね合わせる形でロードして使用するとよいでしょう．

● 使い方 ●

このプログラムが実行されると，画面の左下に“>”が表示されます．これは，いま，カナ大文字の入力モードであることを示しています．

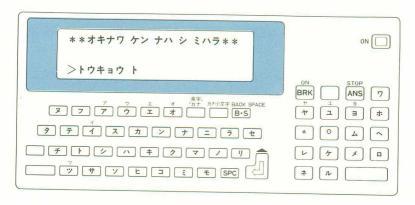
ここで，[⇦]キーを押すと，英文字入力モードとすることができます．このときのメッセージは“<”となります．

カナ大文字入力モードのとき，[⇨]キーを押すと，以後の1文字がカナ小文字入力となり，このときのメッセージは，“=”が表示されます．

なお，[DEL]キーを押すことにより，以前に押された1文字を削除することができます．以上の機能をまとめると，次のようになります．

- カナ大文字入力モード　　>
- 英文文字入力モード　　<
- カナ小文字入力モード　　＝
- 削　除　　DEL

なお，このプログラムの１回の実行で，16文字まで入力することができます．また，必要な文字の入力が終ったら，⏎キーを押すことで，メインプログラムに戻ります．

カナ入力モードでのキー配置

● メインプログラムとのリンク法 ●

例えば，カナで氏名を入力したい事例で考えてみることにします．

配列変数 A$(　)にその氏名をとり込んでいくことにします．

```
10 CLEAR                 ………配列プログラムでは忘れずに！
20 DIM A$(20)
30 GOSUB 9500            ………キャラクタ・データの読み込み
40 FOR B=1 TO 3         ……人数の設定(20人まで自由に変えて下さい)
50 PRINT "No.";B;TAB(3);CHR$(197);CHR
   $(207);CHR$(180);CHR$(32);CHR$(202
   )
60 GOSUB 9000
70 CLS
```

```
 80 PRINT "**";AA$;"**"
 90 A$(B)=AA$ ………カナ文字入力ルーチンでのデータの受け渡し
100 NEXT B
110 FOR B=1 TO 20
120 PRINT A$(B);     ………A$配列の内容を表示する
130 C$=INKEY$:IF C$="" THEN 130
140 IF C$=CHR$(13) THEN 160
150 GOTO 130
160 NEXT B
170 END
```

30行で，9500行～9550行のキャラクタ・データを配列に呼びこんでいます．

40行～100行の**FOR～NEXT**ループで，**A$(　)**の配列に，文字を読み込みます．

170行の**END**は忘れずに付けて下さい．とくに，こうしたサブルーチンを使用するときは，**END**がないと，再び9000行を実行してしまいます．

カナ文字を直接入力できる，次のプログラムを是非有効に使いこなして頂きたいと思います．

カナ文字入力サブルーチンプログラム

```
9000 F=1:AA$="":P=1:LOCATE 0,3:PRINT ">
     ";
9010 V=ASC(INKEY$):IF V=0 THEN 9010
9020 BEEP :IF V<32 THEN 9130
9030 IF P=17 THEN 9010                        ┐カナ大文字
9040 IF F=1 THEN AA$=AA$+W$(V-32):LOCAT       ┘処理
     E P,3:PRINT W$(V-32);:P=P+1:GOTO 9
     010
9050 IF F=-1 THEN 9110
9060 V=ASC(W$(V-32)):F=1:LOCATE 0,3:PRI       ┐
     NT ">";                                   │カナ
9070 IF V>176 THEN IF V<182 THEN V=V-10       │小文字
9080 IF V>211 THEN IF V<215 THEN V=V-40       │処理
9090 IF V=220 THEN V=166                      │
9100 IF V=194 THEN V=175                      ┘
```

```
9110 LOCATE P,3:PRINT CHR$(V);:AA$=AA$+
     CHR$(V)
9120 P=P+1:GOTO 9010
9130 IF V=17 THEN IF P>1 THEN P=P-1:LOC
     ATE P,3:PRINT " ";:AA$=LEFT$(AA$,P
     -1)
9140 IF V=29 THEN F=-F
9150 IF V=28 THEN IF F=1 THEN F=0
9160 LOCATE 0,3:PRINT CHR$(61+F);:IF V=
     13 THEN RETURN  ELSE 9010
9500 DATA 32,205,199,204,199,37,177,39,
     204,177
9510 DATA 206,219,180,205,217,220,200,2
     18,185,210
9520 DATA 222,223,209,212,213,214,181,1
     81,179,179,180,206,64
9530 DATA 193,186,191,188,178,202,183,1
     84,198,207
9540 DATA 201,216,211,208,215,190,192,1
     89,196,182
9550 DATA 197,203,195,187,221,194,91,22
     0,93,219,95,96
9560 DIM W$(90)*1:RESTORE 9500
9570 FOR I=0 TO 64:READ V:W$(I)=CHR$(V)
     :NEXT I
9580 RESTORE 9530:FOR I=65 TO 90:READ V
     :W$(I)=CHR$(V):NEXT I
9590 BEEP :RETURN
```

英文，カナ小文字処理（9110–9120）
バックスペース（削除）（9130）
モード変換（9140）
カナ小文字（9150）
キャラクタデータ入力サブルーチン（9500–9590）

このサブルーチンで，次の変数を使用していますから，メインプログラムでの変数に，ダブリが生じないよう気をつけて下さい．

F：文字のモード設定　I：文字読み込みループ変数　AA$：変換したカナ文字列

P：文字カウンタ　V：読み込んだ文字コード　W$(0)～W$(90)：変換したカナ文字

フローチャートの書き方 3-19

新たにプログラムを作るとき，フローチャート（流れ図）は大変便利で，プログラムを作ってしまった後でも役立ちます．BASICの命令をひととおり覚え，短かいプログラムが作れるようになったら，次には，フローチャートを書く習慣をつけるようにしたいものです。

それによって，長いプログラムでも，処理の範囲や順序をはっきりさせ，手落ち，余分な処理，誤った処理をなくすことができるようになるだけでなく，時間がたってからプログラムを見直したとき，あるいは修正したいときにも大変役立ちます．

■ フローチャート記述のきまり

わが国では，JIS C 6270 情報処理用流れ図記号に定められていますが，これは，国際規格とも矛盾せず対応するように作られています．

JIS では30種類の記号が定められていますが，さし当たり次に示す9種類で十分です．

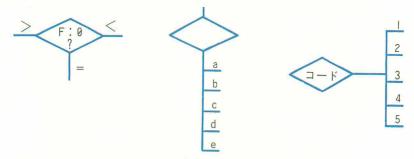
記号	意味
（長方形）	処理（process）—あらゆる種類の処理を表わす．
（ひし形）	判断（decision）—いくつかの経路のうち，どの経路をとらせるかの判断を表わす。
（台形）	手操作入力（manual input）—キーボードからの入力を表わす．
（平行四辺形）	入出力（input／output）—データを入出力する機能を表わす．
（書類形）	書類（document）—プリンタやプロッタに打ち出す
（表示形）	表示（display）—画面に表示する．
○	結合子（connector）—流れ図の他の場所への出口，または他の場所からの入口を示す．
（長円）	端子（terminal）—開始・終了など流れ図の端子を表わす．
―	流れ線（flow line）—記号を結びつける機能を表わす．

■ 記号の使い方

流れの方向は，原則として左から右へ，上から下へとなります．それ以外の方向の場合は，矢印をつけて示します．矢印は，つけた方が分りやすいときはなるべくつけるようにします．

流れ線は，交差しても差しつかえありません．

記号とともに書く文，例えば「スタート」とか「「A＝0」などは，できるだけ記号の中に書くようにします．

一つの記号に二つ以上の出口を書くこと（分岐）は差つかえありませんが，それぞれの出口に分岐の条件を記入する必要があります．

なお，各記号をきれいに，早く書くには，市販されているテンプレートを利用されるとよいでしょう．

■ フローチャートを書く手順

フローチャートは，やりたい仕事の手順をそのまま図式化すればよいのですが，実際にやってみると，なかなかうまくはいきません．そこで，流れをどのように考えるかを，お話しましょう．

フローチャートには，次の２種類があります．

① 概要フローチャート（Logic Flowchart）

② 詳細フローチャート（Detail Flowchart）

どちらも同じ形式で，またほとんど同じ記号を使って書きます．

(1) **概要フローチャート**

フローチャートを作成するに当って，まず実行する仕事の全体を大づかみにして考えます．例えば，次のようです．

データの読み込み──→計算──→書き出し

右図のようになりますが，この概要フローチャートは，詳細フローチャートを作る前段階としての役目とともに，とくに複雑な手順の場合を除けば，これから直ちにプログラム作り（コーディング）に入ることもできます．

(2) **詳細フローチャート**

概要フローチャートに沿って，作業の実行順序を細かく，命令または命令に近い形で書き表わします．ただし，BASIC プログラムでは，あまりに詳細なものは必要ありません．

いずれにしても，プログラムを作る前に，フローチャートを書いて，この段階でじっくり時間をかけることが大切で，準備の手順，入力の仕方，処理の方法，出力の形式へと移っていくようにします．

それでは，例として，２桁の整数を１桁の整数で割る算数ドリルプログラムを，フローチャートから作ってみましょう．

〔概要フローチャート〕

開　始
出題数入力
問題を作る
問題を表示
答を入力させる
正しいか？
出題数完了
NO
YES
終　り

〔詳細フローチャート〕

開　始
出題数入力
100未満の乱数を整数化（A）
A＜10
YES
NO
1以上10未満の乱数を整数化（B）
問題を表示
商を入力
余りを入力
正しいか？
出題数完了
NO
YES
終　り

〔プログラム〕

```
 10 CLS
 20 INPUT "MONDAI NO KAZU";N
 30 FOR I=1 TO N
 40 A=INT(RND*100)
 50 IF A<10 THEN 40
 60 B=INT(RND*9+1)
 70 CLS :PRINT "(";I;" )"
 80 PRINT TAB(5);A;" /";B;"= ";
 90 INPUT "",X
100 PRINT TAB(7);:INPUT " AMARI= ",R
110 IF X<>INT(A/B) THEN 80
120 IF R<>A MOD B THEN 80
130 NEXT I
140 END
```

PB-700のグラフィック機能 3-20

PB－700は，20桁×4行のキャラクターを表示する大型の液晶表示(LCD)をもっていますが，このLCDは，160×32個のドットで構成され，同時にグラフィック表示をも可能にしています．

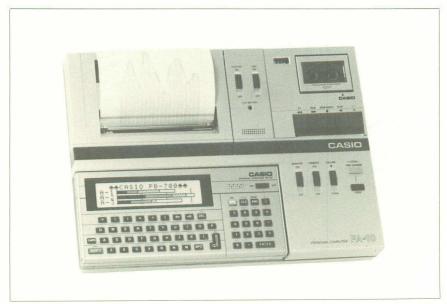
PB－700のグラフィックは，簡単なコマンドで，精密なグラフや図形を描くことができます．

また，PB－700にカセットインタフェイス付ミニプロッタプリンタ(FA－10)を接続すると,紙幅114mm,4色のプロッタ・プリンタを動かすことができます．

紙幅114mmに，最大80桁の文字が書けるということは，本格的なプロッタプリンタとほとんど同じ感覚で使用することができるということです．

PB－700はコンパクトながら，高度なグラフィック機能を備えていますので，これを100％活用できるように，本編で習熟してください．

最初のうちは，多少面倒な感じをもたれるかもしれませんが，少し慣れてくると，グラフィックのおもしろさに必ず取りつかれることでしょう．

カセットインタフェイス付ミニプロッタプリンタ (FA10)

グラフィック命令とLCD座標　3-21

画面上に図形を描くということは，点を次々につないでいくことを意味しています．したがって，指定した位置に点を描くことができればよいわけです．つまり，グラフィック表示をするには，点(ドット)を描いたり，消したりする命令があれば事足りるわけですが，PB−700には，画面グラフィック用に次の２つのコマンドが用意されています．

DRAW　………点や直線を描くコマンド

DRAWC　………点や直線を消すコマンド

また，グラフィック用の便利な関数として，……

POINT　………指定した位置に点が描かれているか否かを示す

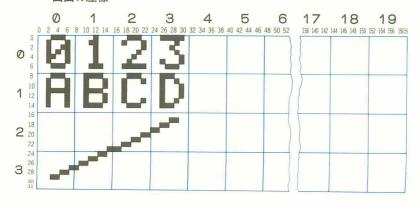
……が用意されています．

さて，これらのコマンドや関数の使い方を説明する前に，まず，画面のドットの位置（座標）について知っておかねばなりません．

下図を見てください．小さく書かれている数字（横：０～159　縦：０～31）が，ドットの位置，つまりグラフィック座標を示しています．また，大きな数字（横：０～19　縦：０～３）が，キャラクター座標です．

キャラクターは，８×８ドットで描かれるグラフィックであると考えることができますが，表示される位置が決められています．

それに対して，グラフィック座標は，どの位置にでも点や線を描くことができます．下図の（３，29）から（29，17）までの直線は，グラフィックコマンドで描いたものですが，どこにでも自由に描くことができます．

画面の座標

前頁で見たように，グラフィック座標は，画面の左上隅を（0，0），右下隅を（159，31）として，X方向160ドット，Y方向32ドット，合計で5120ドットで構成されています．

この座標によって，画面上のドット位置が指定できます．

たとえば，（X，Y）の位置に点を打つ場合は……

DRAW (X,Y)

……とします．また，（X，Y）の点を消すときは……

DRAWC (X,Y)

……とします．

直線を描くときも同じコマンドでよく，直線の両端の座標（X_1, Y_1),（X_2, Y_2）を次のように指定します．

DRAW (X_1, Y_1) − (X_2, Y_2)

これで，（X_1，Y_1)から（X_2，Y_2）までの直線が引かれます．

また，3点（X_1, Y_1)，（X_2, Y_2)，（X_3, Y_3）を結ぶ線を描く場合には……

DRAW (X_1, Y_1) − (X_2, Y_2) − (X_3, Y_3)

……のように指定します．同様に〝−〟で座標を結べば，連続した直線をいくつでも描くことができます．

直線を消すときは……

DRAWC (X_1, Y_1) − (X_2, Y_2)

……のように指定します．

POINT 関数は，指定した位置のドットが点灯，つまり点が描かれているときは，1を与え，点灯していないときは，0を与えます．

たとえば，（X，Y)座標の点を調べるときは……

POINT (X,Y)

……のようにします．

点が描かれているかいないかの値，1，0をプログラム上で利用したいときは，次のように変数に代入して使えばよいでしょう．

A = POINT (X,Y)

前頁で説明した直線の引き方を活用して，まず多角形を描くプログラムを見ていただきましょう．

多角形を描くには，多角形の頂点の座標を，DRAWコマンドで結んでいけばよいわけです．

■三角形を描くプログラム

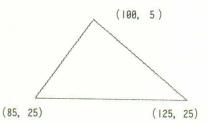
頂点（100，5），（85，25），（125，25）からなる三角形を描くプログラムは，次のようになります．

L-1

```
10 REM --- TRIANGLE ---
20 CLS
30 DRAW(100,5)-(85,25)-(125,25)-(100,
   5)
40 END
```

画面上に描かれる三角形

DRAWコマンドの座標の指定は，数式で指定することもできます．

たとえば，上記のプログラムを数式を使って書き換えてみますと，次のようになります．

L-2

```
10 REM --- TRIANGLE ---
20 CLS
30 X=100:Y=5 ……描き始める点の座標
40 DRAW(X,Y)-(X-15,Y+20)-(X+25,Y+20)-
   (X,Y)
50 END
```

■長方形を描くプログラム

次のプログラムは，(80，5) − (150，28) の直線を対角線とする長方形を描く例です．

L-3

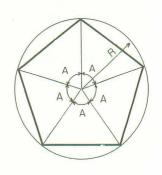
```
10 REM --- RECTANGLE ---
20 CLS
30 DRAW(80,5)-(150,5)-(150,28)-(80,28
   )-(80,5)
40 END
```

■正五角形を描くプログラム

正多角形は，円に内接する多角形で，その頂点が等間隔に並んでいます．

右図は，半径がRの円に内接する正五角形ですが，その5つの頂点と内の中心を結ぶ線は，同じ角度A（この場合は72度）で交わっています．

この性質を利用して，中心（100，18），半径15の内に内接する正五角形のプログラムを書くと，右のようになります．

正五角形

L-4

```
10 REM --- PENTAGON ---
20 CLS
30 R=15            ………半径
40 X=100:Y=18      ………円の中心
50 A=360/5         ………角度
60 FOR I=0 TO 360 STEP A
70 DRAW(X-SIN(I)*R,Y-COS(I)*R)-(X-SIN
   (I+A)*R,Y-COS(I+A)*R)
80 NEXT I
90 END
```

■正N角形を描くプログラム

正五角形のプログラムの角度Aを変更すれば，正N角形を自由に描けます．

正五角形　A＝360／5

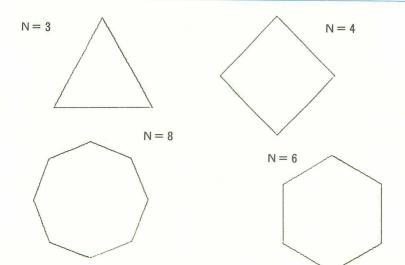
正N角形　A＝360／N

次のプログラムは，Nを入力して，正多角形を描く汎用プログラムです．

ただし，円の中心を（100，15）としてあります．

L-5

```
 10 REM --- POLYGON ---
 20 CLS
 30 R=15                ………半径
 40 X=100:Y=15          ………円の中心
 50 INPUT "N=";N        ………正N角形
 60 A=360/N             ………正N角形の角度
 70 FOR I=0 TO 360 STEP A
 80 DRAW(X-SIN(I)*R,Y-COS(I)*R)-(X-SIN
    (I+A)*R,Y-COS(I+A)*R)
 90 NEXT I
100 END
```

曲線を描く　3-22

曲線を描く場合にも，画面上の座標を指定して，点を描いていけばよいのですが，問題は座標を指定する方法です．

数学の公式を利用すれば，いろいろな曲線を引くことができます．

ここでは，円とサインカーブを描いてみます．

■円を描くプログラム

前項の正多角形のプログラムのNを大きくしていくと，図形は次第に円に近づきます．幾何学的にはどこまでいっても多角形ですが，画面上で曲線を表示するには，この要領で直線をつないで曲線にします．

次のプログラムは，座標（100，15）を中心とする半径15の円を描くものです．

L-6

```
10 REM --- CIRCLE ---
20 CLS
30 FOR I=0 TO 360 STEP 5
40 X=100+COS(I)*15 ………円弧上の点の横の位置
50 Y=15-SIN(I)*15  ………円弧上の点の縦の位置
60 DRAW(X;Y)
70 NEXT I
80 END
```

■サインカーブを描くプログラム

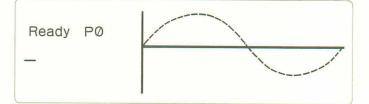
SIN（I）という関数の値は，Iの値が0°から360°まで変わる間に，0→1→0→－1→0のように変化します．

したがって，SIN（I）の値に適当な倍率を乗じたものを，縦の位置として点を描いていけば，サインカーブとなります．

次のプログラムは，サインカーブを描いた後に，X軸，Y軸も描きます．

L-7

```
 10 REM --- SINE ---
 20 CLS
 30 A=65:B=15 ……サインカーブの出発点(X,Y軸の原点)
 40 M=12      …………倍率
 50 FOR I=0 TO 360 STEP 4
 60 X=A+I/4        ………………………X座標点
 70 Y=B-SIN(I)*M      …………Y座標点
 80 DRAW(X,Y)
 90 NEXT I
100 DRAW(A,2)-(A,28)       ………Y軸
110 DRAW(A,B)-(A+92,B)     ………X軸
120 END
```

表示例

折れ線グラフを描く　3-23

グラフにもいろいろありますが，ここでは折れ線グラフを描いてみましょう．

折れ線グラフがよく使われるのは，たとえば，ある期間中の気温の変化とか降水量の変動とか，あるいは株価の足どり，といったように，どちらかといえば，時間的な変化の傾向をつかみたい場合が多いようです．

したがって，できるだけ画面を一杯に使って，変化が明瞭に現れるようにするとよいでしょう．

■月毎の平均気温を折れ線グラフに描くプログラム

ある都市の月毎の平均気温が，次の表の通りであったとします．この気温のデータをDATA文でプログラム中に入力しておいて，READ文で読み出しながら，折れ線グラフに描くプログラムにしてみます．

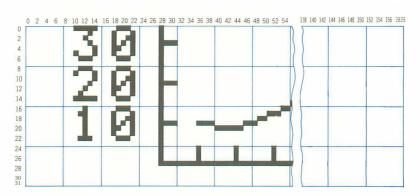
月	気 温℃	月	気 温℃	月	気 温℃
1	11.5	5	19.8	9	22.3
2	9.8	6	23.4	10	18.5
3	13.7	7	26.6	11	15.9
4	18.3	8	28.2	12	14.7

L-8

```
 10 REM---LINE GRAPH---
 20 CLS
 30 FOR I=1 TO 3
 40 PRINT 30-(I-1)*10;CHR$(147)…… =├
 50 NEXT I
 60 PRINT TAB(3);CHR$(154);…… =└
 70 FOR I=1 TO 12
 80 PRINT CHR$(144);…… =┴
 90 NEXT I
100 FOR I=1 TO 12
110 X=36+(I-1)*8
120 READ A
130 Y=4+(30-A)*0.8
140 IF I=1 THEN 160
```

```
150 DRAW(P,Q)-(X,Y)
160 P=X:Q=Y
170 NEXT I
180 DATA11.5,9.8,13.7,18.3,19.8,23.4,2
    6.6,28.2,22.3,18.5,15.9,14.7
200 IF INKEY$="" THEN 200
210 END
```

グラフの数字，縦(Ｙ)軸，横(Ｘ)軸，および折れ線の位置関係がどのようになっているかがわかるように，画面のドットパターンの一部を拡大してみました．図形やグラフを描くプログラムを作るときには，あらかじめこのような絵を描いて，位置関係を調べておくとよいでしょう．

ドットパターン

なお，ちょっとしたプログラムのテクニックですが，200 行がないとプログラムの実行が終ると同時にコマンド待ちとなり，画面に……

Ready P0

_

……のような表示が表われ，せっかく線を描いたグラフがずれてしまいます．

200行目のようなステートメントを入れることにより，何かのキーが押されるまで，プログラムの実行は終了せず，そのままグラフを表示させることができます．

棒グラフを描く準備　3-24

■キャラクタを使った棒グラフの描き方

次のプログラムを実行させ，Nに１～20の数を入力すると，N個だけ▬印を並べて表示します．

L-9

```
10 CLS
20 INPUT "N=";N
30 FOR I=1 TO N
40 PRINT CHR$(131);  ……CHR$(131)は▬印
50 NEXT I
60 END
```

つまり，数Nに比例した長さの棒状の図形が描かれます．

これが棒グラフの原理ですが，画面は20桁しかないので，20を超える量を表現するためには，▬印１個のスケールを小さくすることが必要です．

たとえば，100点満点の点数を表示するには，上記のプログラムの30行を，次のように変えることになります．

```
30 FOR I=1 TO N/6
```

または，……

```
30 FOR I=1 TO N STEP 6
```

しかし，このやり方では，Nがたとえば90から95までが同じ長さとなり，少々荒っぽい表示になってしまいます．

大雑把でもよいときは，このやり方が簡単でよいのですが，もっと精密に描きたいときは，グラフィックを使う方がよいでしょう．

■グラフィックによる棒グラフの描き方

上記の(L―９)のプログラムをグラフィックで描くと，次のページの(L―10)のようになります．

どうですか．この場合は横が160，すなわちキャラクタで描くときの８倍の

L-10

```
10 CLS
20 INPUT "N=";N
30 FOR I=1 TO N*1.5
40 DRAW(I-1,8)-(I-1,14)
50 NEXT I
60 END
```

精度で表示できます.

このプログラムでは，ベタ塗りの棒ですが，棒の模様を変えることによって，より見易いプログラムを書くことができます.

次のⓐ～ⓑのプログラムを，（L－10）のプログラムにそれぞれ追加あるいは変更することによって，次のような模様で，棒グラフを描くことができます.

ⓐ　ⓑ　ⓒ　ⓓ

ⓐ
```
35 IF I=N*1.5 THEN 40
36 IF I MOD 2=0 THEN 50
```

ⓑ
```
30 FOR I=8 TO 14 STEP 2
40 DRAW(0,I)-(N*1.5-1,I)
```

ⓒ
```
35 FOR J=8 TO 14 STEP 2
40 DRAW(I-1,J+(I+1) MOD 2)
45 NEXT J
```

ⓓ
```
25 FOR J=8 TO 14
30 FOR I=1 TO N*1.5 STEP 3
35 X=(J-8) MOD 3+I-1
36 IF X>N*1.5-1 THEN X=N*1.5-1
40 DRAW(X,J)
50 NEXT I:NEXT J
```

棒グラフプログラム2例　3-25

棒グラフは，たとえば，生徒やセールスマンなどの成績，生産高，商品別売上高のように，相対的な関係を見るときに多くつかわれます．

ただ，相対的な比較ばかりではなく，各データの大きさ，つまり棒の長さもある程度正確に表示する必要もありますので，画面の限られた範囲内で，適当なスケールで描くことが大切です．

棒グラフにも，目的によっていろいろの描き方がありますが，基本的なものとして，１本の棒で単一の量を示すもの（たとえば，製品別の生産高を示す場合）と，１本の棒で全体の量と同時に，その内訳をも示すもの（たとえば，総生産高と製品別の割合を示す場合）などがあります．

ここでは次の例を使って，この２種類の棒グラフを描くプログラムを示します．

データ例：ある自動車メーカーのＡ，Ｂの２種類の生産台数が，次表のようであった．

	1980年	1981年	1982年
Ａ車	48200台	57200台	67200台
Ｂ車	39200台	31100台	27500台

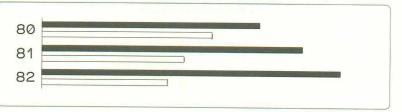
Ａ車の棒グラフの形： ■

Ｂ車の棒グラフの形： □

■１本の棒で単一の量を示す棒グラフのプログラム

L-11

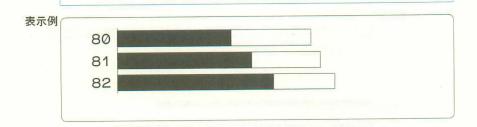
```
10 REM --- BARGRAPH ---
20 CLS
30 FOR I=0 TO 2
40 LOCATE 0,I
50 PRINT 80+I;CHR$(136)  ……CHR$(136)=(|)
60 FOR J=1 TO 2
70 READ A
80 FOR K=0 TO 2
```

```
 90 Y=I*8+(J-1)*4+K
100 IF J=2 THEN IF K=1 THEN DRAW(24+A/
    500,Y):GOTO 120
110 DRAW(25,Y)-(24+A/500,Y)
120 NEXT K:NEXT J:NEXT I
130 IF INKEY$="" THEN 130
140 END
150 DATA48200,39200,57200,31100,67200,
    27500
```

■1本の棒で全体の量と同時に内訳をも示す棒グラフのプログラム

L-12

```
 10 REM --- BARGRAPH ---
 20 CLS
 30 FOR I=0 TO 2
 40 LOCATE 0,I
 50 PRINT 80+I;CHR$(136)
 60 READ A,B
 70 FOR J=0 TO 6
 80 Y=I*8+J
 90 DRAW(25,Y)-(24+A/1000,Y)
100 NEXT J
110 X=24+A/1000:Y=I*8
120 DRAW(X+1,Y)-(X+B/1000,Y)-(X+B/1000
    ,Y+6)-(X+1,Y+6)
```

```
130 NEXT I
140 IF INKEY$="" THEN 140
150 END
160 DATA48200,39200,57200,31100,67200,
    27500
```

表示例

動画(アニメーション)の描き方　3-26

■絵を動かすしくみ

次のプログラムを実行すると，＊印が左右に動きます．

L-13

```
 10 CLS
 20 FOR I=5 TO 15          ┐
 30 LOCATE I,1             │ 左から右へ移動
 40 PRINT " *"             │
 50 NEXT I                 ┘
 60 FOR I=15 TO 5 STEP -1  ┐
 70 LOCATE I,1             │ 右から左へ移動
 80 PRINT "* "             │
 90 NEXT I                 ┘
100 GOTO 20
```

プログラムを見ると，20行で制御変数Iを5から15まで変化させながら，30行のステートメントでカーソルの位置が，(5，1)から(15，1) まで変ります．その位置に〝␣＊〟を描くことによって，＊印を描くと同時に先に描かれていた＊印をスペースによって消します．

これを繰り返すことによって，左から右へ動いていくように見えます．

また，60行～90行は，逆に右から左へ動かすステートメントです．

実は，これがグラフィックアニメーションの原理です．

つまり，描くことと消すことを組み合せることによって，動いているように見せるわけです．

縦に動かすのも理屈は同じですが，描くことと消すことを同時に行なうわけにはいかず，次のようなプログラムになります．

L-14

```
10 CLS
20 FOR I=0 TO 3
30 LOCATE 10,I:PRINT "*";
40 IF I=0 THEN 60                ……上から下へ移動
```

```
 50 LOCATE 10,I-1:PRINT " "
 60 NEXT I
 70 FOR I=3 TO 0 STEP -1
 80 LOCATE 10,I:PRINT "*";
 90 IF I=3 THEN 110
100 LOCATE 10,I+1:PRINT " "
110 NEXT I
120 GOTO 20
```

上から下へ移動(50〜60)
下から上へ移動(70〜110)

■スピードの変え方

前述のプログラムでは，＊印の移動が速すぎて困るような場合，スピードの調節を FOR～NFXT 文を使って行ないます．

(L－13)のプログラムに，次の行を追加して実行してみてください．

```
45 FOR J=1 TO 50:NEXT J
```

このステートメントによって，左から右へ移動するスピードが少し遅くなります．この FOR～NEXT 文の終値を小さくすれば速くなり，大きくすれば遅くなります．

このようなステートメントを必要な箇所に，適当に追加していけばよいわけです．

■点を曲線的に動かす

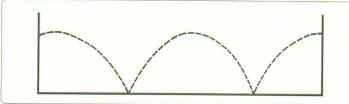
曲線上を動かすときは，グラフィック座標にする方が，動きが滑らかになります．

次のプログラムは，実行例のように，枠内の点線のような軌跡上を点が繰り返し移動します．

L-15

```
10 CLS
20 DRAW(13,0)-(13,31)-(137,31)-(137,0)
         ……枠を描く
```

```
 30 DRAW(14,0)-(14,30)-(136,30)-(136,0
          ………枠を描く
 40 N=1:P=14
 50 FOR I=0 TO 360 STEP 3
 60 X=P+N:Y=29-25*ABS(COS(I)) ……曲線の座標
 70 DRAW(X,Y)                      点の計算
 80 IF P=14 THEN 100
 90 DRAWC(P,Q)
100 P=X:Q=Y
110 NEXT I
120 N=-N
130 GOTO 50
```

実行例

■POINT 関数の使い方

POINT 関数は，グラフィック座標の指定した位置に，点（ドット）が描かれているかどうかをチェックするものです．

もし，横座標(X)，縦座標(Y)の位置，つまり（X，Y）に点がある場合は，

POINT (X, Y) … 1

……無い場合は，

POINT (X, Y) … 0

……となります．

次のプログラムは，POINT 関数の使用例ですが，まず最初に，画面の第1行に“CASIO　PB−700”と書き，続いて POINT 関数で各ドットの有無をチェックし，与えられた0，1の値にもとずいて，第3行へ書き写すものです．

L-16

```
 10 CLS
 20 PRINT "*** CASIO PB-700***"
 30 FOR X=0 TO 159
 40 FOR Y=0 TO 7
 50 IF POINT(X,Y)=0 THEN 70
 60 DRAW(X,Y+16)
 70 NEXT Y
 80 NEXT X
 90 IF INKEY$="" THEN 90
100 END
```

30行～80行で，キャラクタ座標の第1行目のドットの有無を全部調べ，50行で，もしドットが打たれていなければ，70行へ飛び，打たれていれば，60行で第3行目の同じ位置にドットを打っています．

もし，プロッタプリンタ（FA−10）があれば，画面のハードコピーをとることができます．プロッタプリンタの使い方は，別の項で説明しますが，参考のために，ハードコピーのプログラムを示しておきます．

L-17

```
 10 CLS
 20 PRINT "CASIO PB-700"
 25 LPRINT CHR$(28);CHR$(37)
 30 FOR X=0 TO 159
 40 FOR Y=0 TO 7
 50 IF POINT(X,Y)=0 THEN 70
 55 V=X*.59:W=Y*.59
 60 LPRINT "D";V;",";-1*W;",";V+0.4;",
    ";-1*(W+0.4)
 70 NEXT Y
 80 NEXT X
 90 IF INKEY$="" THEN 90
100 END
```

ゲームへの応用　3-27

アニメーションは，しばしばゲームに使用されます．

次のプログラムは，ブロックくずしのプログラムです．特にむずかしいテクニックはないのですが，画面の変化をいかにもそれらしく見せるための技法が要求されます．

初めての方には，少しむずかしいかもしれませんが，ブロックの位置をもう少し右に動かすとか，ラケットをもう少し左にやるといったように，少し工夫をしてみてください．

それによって，みなさんのゲーム作り，いやプログラム作りの腕がメキメキ上達することは確かです．

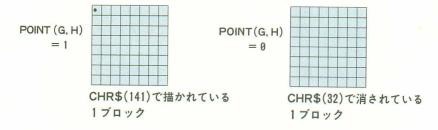
まず，プログラムを打ち込んでください．

L-18

```
 10 REM ---BLOCK KUZUSHI ---
 20 CLS
 30 FOR I=0 TO 3
 40 LOCATE 2,I:PRINT CHR$(137);
 50 NEXT I
100 FOR I=3 TO 6
110 FOR J=0 TO 3
120 LOCATE I,J
130 PRINT CHR$(141);
140 NEXT J:NEXT I
200 LOCATE 17,1
210 PRINT CHR$(147)
220 R=1:S=0
300 FOR N=3 TO 1 STEP -1
310 LOCATE 0,0:PRINT N
320 X=10:Y=INT(RND*2+1):A=1:B=1
330 LOCATE X,Y:PRINT CHR$(236);
340 Q$=INKEY$
350 IF Q$="1" THEN GOSUB 500 ELSE IF Q
```

```
    $="0" THEN GOSUB 600
360 IF X=3 THEN A=-A
370 IF Y=0 THEN B=-B ELSE IF Y=3 THEN
    B=-B
380 IF X=16 THEN GOSUB 800
390 LOCATE X,Y:PRINT " ";
400 IF X>16 THEN 440
410 G=(X+A)*8:H=(Y+B)*8
420 P=POINT(G,H)
430 IF P=1 THEN GOSUB 700
440 X=X+A:Y=Y+B
450 IF X>18 THEN BEEP :BEEP :BEEP  ELS
    E 330
460 NEXT N
470 CLS :LOCATE 2,1:PRINT "SCORE ";S
480 FOR I=1 TO 6:BEEP 1:NEXT I
490 IF INKEY$="" THEN 490 ELSE END
500 LOCATE 17,R:PRINT " ";
510 R=R-1
520 IF R<0 THEN R=0
530 LOCATE 17,R:PRINT CHR$(147);
540 RETURN
600 LOCATE 17,R:PRINT " ";
610 R=R+1:IF R>3 THEN R=3
620 LOCATE 17,R:PRINT CHR$(147);
630 RETURN
700 S=S+8-X:BEEP 1
710 LOCATE X+A,Y+B:PRINT " ";
720 X=X+A:Y=Y+B:A=-A
730 IF Y=0 THEN B=-B ELSE IF Y=3 THEN
    B=-B
740 RETURN
800 IF Y=R THEN A=-A:BEEP  ELSE IF Y+B
```

```
     =R THEN BEEP :A=-A ELSE 830
810 IF RND<0.7 THEN 830
820 LOCATE X,Y:PRINT " ";:X=X-1
830 RETURN
```

420 行の POINT 関数は，下図のように，1ブロックの左上隅のドットの有無で1ブロック全体の有無をチェックしています．

■ゲームのやり方

このプログラムを RUN すると，ブロックとラケットが描かれ，玉が斜に動き出します．

ラケットよりも玉が右に行ってしまうと，1回終了になり，合計3回プレイできます．

ラケットは〝1〞を押すと上がり，〝0〞を押すと下がります．

3回のプレイが終わると得点が表示されます．

■ブロックくずしプログラム変数リスト

変数	内容
A，B	ボールの移動の方向
G，H	ブロックの有無を判定する座標
I，J	ブロック，壁を書くための変数
N	プレイできる残りの回数
P	ブロックの有無
Q$	ラケットの移動方向
R	ラケットの位置
S	スコア
X，Y	ボールの位置

プロッタプリンタで図形を描く　3-28

PB−700にカセットインターフェイス付ミニプロッタプリンタを接続すると，プロッタが使えます．プロッタは，プリンタとしても使えますが，ドットプリンタが点を打つことによって，字や図形を表すのに対して，プロッタは点と点を結ぶ線によって表わしますので，図形を描くのに適しています．

画面に図形を描くのとほとんど同じ要領で描けるのですが，なにぶん画面が電気的に描くのに対し，プロッタは機械的に描くことになりますので，スピードを上げる必要があります．

そのために，画面とは異なるプロッタ専用のコマンドが用意されています．

コマンド一覧を次のページに示しておきます．

かなり多くのコマンドがあるため，ちょっと面倒だと思われるかもしれませんが，これだけたくさんのコマンドが用意されているために，図形を描くスピードが速くなり，そればかりではなく，プログラミングの手間が大幅に省けるのです．

■命令の伝え方

プロッタでグラフィックを行なう場合の命令は，すべてLPRINTで始まります．LPRINTにつづいて，次のページのようなコマンドを書くことによって，いろいろな働きをします．

ただし，これらのコマンドを使う前には，必ず次の命令でグラフィックモードの指定をしておかなければなりません．

グラフィックモードの指定：LPRINT　CHR$(28)；CHR$(37)

この命令は，このまま覚えておくと便利です．

また，グラフィックモードの解除，つまりキャラクタを印字するキャラクタモードにするには，次の命令を使います．

キャラクタモードの指定：LPRINT　CHR$(28)；CHR$(46)

これも覚えておくとよいでしょう．

表1　プロッタコマンド一覧表

	コマンド	名　称	説　明
作図用	O	ORIGIN	ORG 座標の原点指定
	D	DRAW	ORG 座標で指定された点と点を結ぶ
	I	RELATIVE DRAW	変化量で示された点まで線を引く
	M	MOVE	ORG 座標で示された点まで，ペンアップで移動
	R	RELATIVE MOVE	変化量で示された点までペンアップで移動
	A	QUAD	ORG 座標で示された2点を対角とするX軸，Y軸に平行四辺形を描く
	C	CIRCLE	ORG 座標で指定された点を中心とする円，円弧を描く
	X	AXIS	ORG 座標の原点より，＋Y，＋X，－Y，－X方向に座標軸を描く
	G	GRID	指定された四角の中に横縞，縦縞を描く
	L	LINE TYPE	実線，破線，一点鎖線，二点鎖線を描く
	B	LINE SCALE	破線，一点鎖線，二点鎖線のピッチの指定
文字・記号用	S	ALPHA SCALE	文字記号の大きさを指定
	Q	ALPHA ROTATE	文字，記号の回転方向を指定
	Z	SPACE	次桁，次行の文字間隔を指定
	Y	YOKO	横書き，縦書きの指定
	P	PRINT	文字列の印字
	N	MARK	ペン位置を中心にマークを描く
制御用	J	NEW PEN	ペンの色を選択
	F	LINE FEED	行単位の紙送り，紙もどし
	H	HOME	絶対座標の変更，または図形を見やすい位置へ移動
	@	TEST	ペンならし，ペンのインク状態のチェック〈実行例〉LPRINT CHR$(28)；CHR$(37)；CHR$(64) ↵
文字制御	T	TAB	タビュレーション
	?	FORMAT	プログラムリスト出力用

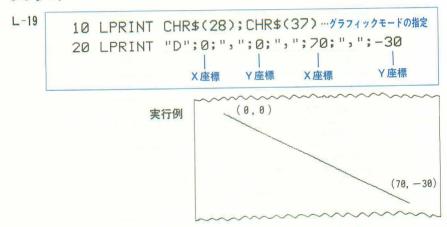
直線を引く　3-29

■ 線を引く（Dコマンド）

直線を引くコマンドが，Dです．

プロッタの紙幅は114㎜ありますが，実際に図形を描ける紙幅は96㎜です．

プロッタの電源をONにすると，プロッタのペンは左端に位置します．この点を（0，0）の座標としますと，そのラインの右端の座標は（96，0）となります．この（0，0）から（96，0）の線をX軸としますと，ラインより上側は，Y軸のプラスの座標，下側はマイナスの座標となります．

では，Dコマンドを使って，（0，0）から（70，−30）まで線を引いてみましょう．

このように，LPRINTに続くコマンド「D」やカンマ「，」のようなキャラクタや記号は，すべて引用符「″」でくくるとともに，それらの間の区切りには，必ずセミコロン「；」をつけます．

一般に，点（X1，Y1）から（X2，Y2）まで直線を引く命令は……

LPRINT "D";X1;",";Y1;",";X2;",";Y2

㊟ パラメーターが数値の場合，例題の20行は次のようにも書けます．

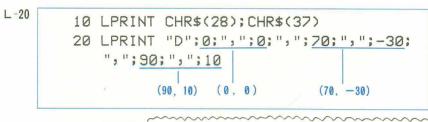
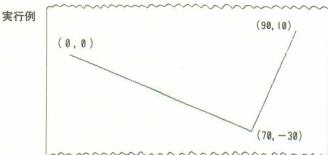
```
20 LPRINT "D0,0,70,-30"
```

他のコマンドでも試して見て下さい．

……で表わします．

では次に，（0，0）−（70，−30）−（90，10）の2本の線を引いてみます．

L-20

実行例

このように，入力可能な文字数（79文字）の範囲内で，いくつの座標点を指定してもよく，連続したいくつもの線をひくことができます．

また，行をかえて次のように書くこともできます．

L-21

```
10 LPRINT CHR$(28);CHR$(37)
20 LPRINT "D";0;",";0;",";70;",";-30
30 LPRINT "D";",";90;",";10
```

現在ペンのある位置から(90, 10)まで線を引く

一般に，現在ペンのある位置から（X2, Y2）まで線を引くときは……

```
LPRINT "D";",";X2;",";Y2
```

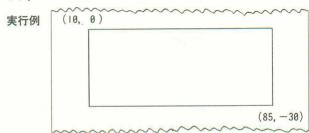
……とします．

引用符やカンマ，セミコロンがたくさん使われて，少々面倒な感じもしますが，慣れればそれほど厄介に感じませんし，繰り返しが多いので，むずかしいことではありません．

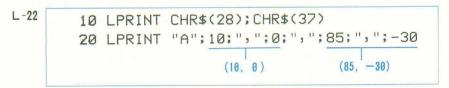
長方形と円と色の指定　3-30

■長方形を描く（Aコマンド）

下図の長方形は，点(10,　0)と点(85,　−30)の対角線の両端を指定し，Aコマンドで描いたものです．

実行例

プログラムは，Aコマンドを使って，次のようになります．

L-22

```
10 LPRINT CHR$(28);CHR$(37)
20 LPRINT "A";10;",";0;",";85;",";-30
```


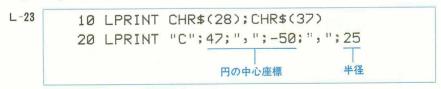
このように，対角線の両端の座標を指定するだけでよいのです．

■円を描く（Cコマンド）

円は，Cコマンドを使って，中心座標と半径を指定するだけで描けます．

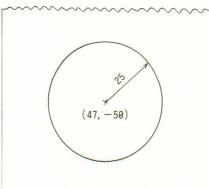
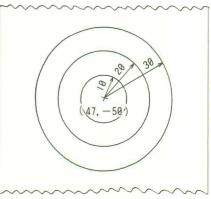
紙のほぼ中央の点(47, −50)を中心に，半径25の円を描くプログラムは，次のようになります．

L-23

```
10 LPRINT CHR$(28);CHR$(37)
20 LPRINT "C";47;",";-50;",";25
```


画面の上に描くプログラムと比較すると，ずい分簡単ですね．

このプログラムの実行例は，次のページの左上図のようになります．

実行例

実行例

次に，中心座標が（47, −50）で，半径が10, 20, 30の同心円を描いてみましょう．プログラムは，次のようになります．また，実行例は右上図のようになります．

L-24

```
10 LPRINT CHR$(28);CHR$(37)
20 FOR R=1 TO 3
30 LPRINT "C";47;",";-50;",";R*10
40 NEXT R
```

半径を変える

■色の指定（Jコマンド）

プロッタには，黒，青，緑，赤の4色のペンが内蔵されていますが，この4色を使い分けることができます．

ためしに，前項のプログラムに25行を追加した次のプログラムを実行してみてください．同心円が色分けされて描かれます．

L-25

```
10 LPRINT CHR$(28);CHR$(37)
20 FOR R=1 TO 3
25 LPRINT "J";R
30 LPRINT "C";47;",";-50;",";R*10
40 NEXT R
```

このように，色の指定はＪコマンドで行ないますが，それぞれの色は，次のような値で得られます．

LPRINT "J";Ø ………黒
LPRINT "J";1 ………青
LPRINT "J";2 ………緑
LPRINT "J";3 ………赤

また，上記のように値を数値で与えるときは，"JØ" "J1" "J2" "J3" のように書くことができますが，変数や数式のときには，セミコロン（；）で区切って指定しなければなりません．

グラフを描くテクニック　3-31

プロッタは，折れ線グラフ，棒グラフ，円グラフなど，各種のグラフを描くとき最高に威力を発揮します．

■座標を決める（O，H，Xコマンド）

グラフには座標がつきものですが，最初にやることは位置決めです．

LPRINT "O"；X；","；Y

Oコマンドによって，（X，Y）の位置に座標の原点を置くことができます．

LPRINT "H 20"　または　LPRINT "H"；A（Aは変数）

この命令によって，原点の位置が，20あるいは変数Aの値だけ下方に移動します．

以上のような命令で原点が決まったら，折れ線グラフや棒グラフなどの座標軸（X軸，Y軸）と目盛を描くことになりますが，それには非常に便利なXコマンドを使います．

LPRINT "X"；0；","；5；","；10
LPRINT "X"；1；","；5；","；16

Xのすぐ後の数字は，軸の方向を示します．

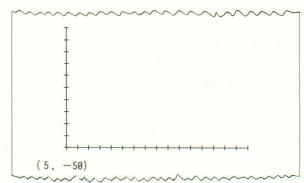
0：Y軸を原点から上に向って引く
1：X軸を原点から右に向って引く
2：Y軸を原点から下に向って引く
3：X軸を原点から左に向って引く

次の数値5は，目盛のピッチ（単位：mm）を指定し，その右の数値10や16は，目盛の数を示します．

これらのコマンドを利用した例が，次のプログラムです．

L-26

```
10 LPRINT CHR$(28);CHR$(37)
20 LPRINT "O5,-50"       ……座標の原点指定
30 LPRINT "X0,5,10"      ……Y軸と目盛を描く
40 LPRINT "X1,5,16"      ……X軸と目盛を描く
```

このプログラムを実行すると，（5，−50）を原点とする次のような座標軸が描かれます．

■線の種類を変える（L，Bコマンド）

グラフで何種類ものデータを扱う場合，何かの方法で類別する必要があります．その1つの方法として，前に述べた色を変える方法があります．

これは非常に有効な方法ですが，欠点としてコピーしたとき区別できなくなることです．

このようなときは，線の種類を変える方法と，棒グラフなどでは，棒の模様を変える方法があります．

① 線の種類を指定する（Lコマンド）

L-27

```
10 LPRINT CHR$(28);CHR$(37)
20 FOR I=0 TO 3
30 LPRINT "L";I
40 LPRINT "D20,";-I*5;",70,";- I*5
50 NEXT I
```

このプログラムを実行させると，次のような4種類の線を引きます．

———————————— 実線

— — — — — — — — 点線

—-—-—-—-—-—-— 一点鎖線

—--—--—--—--— 二点鎖線

つまり，Lの後の0〜3の値は，次の意味をもちます．

"L"；0　………　実線
"L"；1　………　点線
"L"；2　………　一点鎖線
"L"；3　………　二点鎖線

② 点線，鎖線のピッチを変える（Bコマンド）

L-28

```
10 LPRINT CHR$(28);CHR$(37)
20 FOR I=0 TO 10
30 LPRINT "L1":LPRINT "B";I
40 LPRINT "D20";",";-I*5;",";70;",";-
   I*5
50 NEXT I
```

このプログラムを実行すると，次のようなピッチの異なる11本の線を引きます．

コマンドBで指定したIの値がピッチ（単位：mm）になっており，線の長さはその1/2です．ただし，Iが0のときに限り，ピッチは0.5mmとなります．

なお，ピッチは1mm以上で0.2mmの刻みで変えることができ，Bコマンドの指定がないときは，6.4mmにセットされます．

■縞模様を描く（Gコマンド）

L-29

```
10 LPRINT CHR$(28);CHR$(37)
20 FOR I=0 TO 2
30 LPRINT "M20";",";0
40 LPRINT "G";I;",";50;",";-20
50 LPRINT "A20";",";0;",";70;",";-20
60 LPRINT "H10"
70 NEXT I
```

このプログラムの40行に，Gコマンドが使われています．

変数Iは0から2まで変化しますが……

0：無地

1：横縞

2：縦縞

……の3種類を指定することができます．

それに続く数値50および−20は，模様を描く範囲（現在のペンの位置から右に50mm，下に20mm）を指定しています．

さらに，その後に「; "," ; 2」を加えると，縞のピッチが2mmになります．つまり……

LPRINT "G" ; 1 ; "," ; 50 ; "," ; −20 ; "," ; 2

模様の種類　右へ50mm　下へ20mm　2mmピッチ

……となります．

棒グラフをこのような模様分けをすると，より見易くなります．

■ペンを移動する（Mコマンド）

前項のプログラム30行に，「M」という新しいコマンドが出てきましたのでついでに説明しておきます．Mコマンドは……

LPRINT "M" ; 20 ; "," ; 0 または "M20,0"

……のように書きますが，これはペンの位置を（20，0）へ移動せよ，という命令です．つまり，移動するだけで何も描きません．

■円グラフを描く（Cコマンドの応用）

前項までで，折れ線グラフや棒グラフを描く基本を説明したわけですが，円グラフは初めて出てきますので，説明を加えておきます．

円を描くのは，すでに述べたように，Cコマンドを使いますが，円グラフはデータの大きさに応じて，扇形に区切る線を引かねばなりません．

たとえば，100，200，300，400，500の割合に分割するプログラムを示しておきましょう．

```
 10 LPRINT CHR$(28);CHR$(37)
 20 LPRINT "C48,-50,30"
 30 S=0:T=0
 40 FOR I=1 TO 5:READ A:T=T+A:NEXT I:R
    ESTORE :A=0
 50 FOR I=1 TO 5:A1=A
 60 READ A:S=S+A
 70 X=48+INT(30*SIN(360*S/T)):Y=-50+IN
    T(30*COS(360*S/T))
 80 LPRINT "D48,-50,";X;",";Y
 90 A2=(A1+A)/2:GOSUB 200:NEXT I
 95 END
100 DATA100,200,300,400,500
200 X=50+INT(20*SIN(360*S/(T+A2))):Y=-
    48+INT(20*COS(360*S/(T+A2)))
210 LPRINT "M";X-10;",";Y
220 B$=STR$(A)
230 LPRINT "P";B$
240 RETURN
```

実行例

40行目でデータの統計を求め，60～70行目で，各データの比率と分割する円周上のX，Y座標を求めています．

そして，80行目で求められたX，Y点から中心への線を引いています．

第4章に入る前に

■第4章で解説する各コマンドの「書式」の部分にでてくる，次の用語について，以下に説明しておきます．

●「数　　式」—数値，変数，計算式を表わします．計算式では演算子の優先順位に従って計算を行ないます．

●「文 字 式」—文字定数，文字変数，文字列等を表わします．文字列は“＋”符号により連結することができます．つまり文字列のたし算が文字式ということになります．

〔例〕“ABC”＋“DE”＋“F”＝“ABCDEF”

ただし，＋で連結した文字列は，79文字を超えることはできません．

●「変　　数」—変数はデータを格納しておくものです．データには数値と，文字・記号がありますので，変数にも，数値変数（A，B…など）と文字変数（A$, B$‥‥など）の2種類があります．詳しくは第3章をごらん下さい．

●「変 数 名」—変数はA～Zの英大文字を先頭に，$，数字，英大文字などが続いて構成されます．

〔例〕 A, A$, AB, A1, XY$, X1$……など

変数名とは，このような変数の形をいいます．

●「条 件 式」—関係演算子(＝，＝<，>，<>，など)で，左辺と右辺の大小を比較する関係式です．

●「行 番 号」—プログラムの先頭につける番号のことで，1～9999の範囲で使用できます．

●「注 釈 文」—プログラムリストを見るとき，内容がわかるように，プログラム中の要所要所に書き込んでおくものです．プログラムの実行にはいっさい影響しません．

●「メッセージ」—入力を正確に，また出力をわかりやすくするために，画面に表示させる文で，“　”で囲って使います．

●「ファイル名」—プログラムやデータを，カセットテープと本体間でやりとりする場合，名前をつけた方がわかりやすくなります．その名前のことをいいます．

第4章
コマンド・リファレンス

PB-700コマンド・文・組み込み関数一覧

音
ベル
BEEP
入出力文
画面への出力文
出力
PRINT
書式制御出力
PRINT USING
キーボードからの入力文
入力
INPUT
1文字入力
INKEY$
カセットファイルへのデータ入出力文
カセットファイルの入出力
出力
PUT
入力
GET
トレース文
トレースモード設定
TRON
トレースモード解除
TROFF
組み込み関数
表示用関数
画面情報
点の有無
POINT
書式制御
水平位置の指定
TAB
数値関数
数学関数
正弦
SIN
余弦
COS
正接
TAN
逆正弦
ASN
逆余弦
ACS
逆正接
ATN
角度単位の設定
ANGLE
指数関数
EXP
平方根
SQR
自然対数
LOG
常用対数
LGT
絶対値
ABS
整数化
INT
小数部
FRAC
符号
SGN
四捨五入
ROUND
円周率
PI
乱数発生
RND
文字関数
文字↔ASCIIコード
ASCIIコード
ASC
文字変換
CHR$
文字列↔数値
数字文字変換
STR$
文字数値変換
VAL
文字の抽出
文字列の左の部分
LEFT$
文字列の抜き出し
MID$
文字列の右の部分
RIGHT$
文字の位置・文字数
文字列の長さ
LEN

マニュアル・コマンド

CONT　(continue：コンティニュー)

機　能	STOP文もしくは STOP/ANS キーにより停止したプログラムの実行を再開する
書　式	CONT

プログラムの作成とは，なかなか難しいもので，完璧に思えるプログラムでも，最初からこちらの期待通りに動くというわけにはなかなかいきません．そこで，デバッグを重ねて徐々に完全なプログラムに仕上げていくわけですが，そんなとき，このCONT命令が便利になります．

解　説

CONT命令は，プログラム中のSTOP命令や STOP/ANS キーの操作で，プログラムの実行がSTOP状態になっているとき，これを再開するために使うコマンドです．再開は，もちろんSTOPコマンドで停止した行の次の文から行なわれます．

使い方

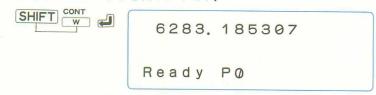
プログラムの作成時，デバッグを容易にするために実行の節目にSTOPコマンドを挿入しておきます．

例えば，次のようなプログラムを作ったとします．

```
10 READ R,H
20 S=PI*R^2
30 STOP
40 V=S*H
50 PRINT R,H,S,V
60 DATA10,20
70 END
```

このプログラムをRUN命令により実行させますと，30行目のSTOPコマンドにより実行を中止して，次のような表示になります．

ここで，10行目，20行目の実行内容をチェックしてみます．そのために，マニュアルで変数を調べてみましょう．R ENTER，H ENTER，S ENTER で変数内容を表示させると，それぞれ10，20，314.1592654 が現われ，正しく実行されていることがわかります．

このようにしてチェックが終了したら，続いてそれ以降のプログラムの実行に移りますが，ここでCONT命令を使うわけです．

このように，40行目以降を実行し，最終結果Vの値を表示して終了します．こうして，プログラムを実行単位ごとにSTOP命令で区切り，変数内容をチェックしていけば，デバッグが非常に容易になります．

なお，STOPコマンドでプログラムの実行が停止した状態で，EDIT モードにしてプログラムの訂正などを行なった場合は，CONT命令を使っても再開しません．このようなときは，RUN行番号で実行を再開して下さい．

参 照　STOP

マニュアル・コマンド

DELETE　(delete：デリート)

機　能	プログラムを行単位で部分的に削除する
書　式	DELETE ln DELETE ln− DELETE −lm DELETE ln−lm ln：削除開始行　lm：削除終了行 (1≦ln≦lm≦9999)

解　説

このコマンドは，プログラム中の特定の行，または特定範囲の行を削除するために使われます．

DELETEの使い方には以下のような基本型があります．

① DELETE　行番号ln　……行番号lnを削除する
② DELETE　行番号ln−……行番号ln以降の行を削除する
③ DELETE　−行番号ln……行番号ln以前の行を削除する
④ DELETE　行番号ln−lm…行番号lnからlmの間の行を削除する

ただし，ln，lmには次のような制約があります．

0≦ln≦lm≦9999

このコマンドは，マニュアルコマンドですから，プログラム中で使用することはできません．

使い方

では，次のプログラムを使って，もう少し詳しく説明してみましょう．

```
10 REM DELETE SAMPLE
20 PRINT "20:A"
30 PRINT "30:B"
40 PRINT "40:C"
50 PRINT "50:D"
60 PRINT "60:E"
```

```
 70 PRINT "70:F"
 80 PRINT "80:G"
 90 PRINT "90:H"
100 PRINT "END"
110 END
```

① DELETE 行番号 ln

例えば，上のプログラム中，50行目を削除したい場合には，次のキー操作で可能です．

DELETE　50 ↵

なお，これと同じことが，次の操作でもできます．

50 ↵

② DELETE 行番号 ln－　（－：ハイフンは，マイナスキーを使う）

先のプログラムで，70行目から最後までを削除したい場合，次のキー操作を行ないます．

DELETE　70－ ↵

また，

DELETE　65－ ↵

でも同様に，70行目以降を削除してしまいます．つまり，65行目がない場合には，それ以降で最も近い行番号70から削除を行なうというわけです．

③ DELETE－行番号 lm

これは，②とは逆に行番号lm以前を頭から削除するものです．例えば次のキー操作を行なって下さい．

DELETE　－40 ↵

SHIFT LIST/L ↵ でリストを見てみますと，10～40行目が削除されているはずです．なお，次のようにしても，全く同じ結果が得られます．

DELETE　－42 ↵

4 DELETE 行番号 ln－行番号 lm

先のプログラムで，30行目から60行目までを削除する場合，次のキー操作を行ないます．

DELETE　30－60⏎

☞ この場合，lnとlmを逆にして，次のような操作をすると，プログラム全体をこわしてしまいますので十分注意が必要です．

DELETE　60－30⏎　……**プログラムがこわれます**

■パスワード付のプログラムには使用できません．

DELETEコマンドは，PASSコマンド（148 ページ参照）によってロックされているプログラムに対しては，PRエラーを表示して実行されません．

■行番号に小数部を含む場合は，SNエラーとなり実行されません．

参 照　NEW，NEW ALL

マニュアル・コマンド

EDIT (edit：エディット)

機　能	プログラムを修正可能状態にする
書　式	EDIT EDIT ln　（0≦ln≦9999）

解　説

EDITコマンドはプログラムの削除，追加，訂正など，いわゆる編集をするときに使うものです．

EDIT ⏎ で，プログラムの先頭の行が表示され，末尾にカーソルが現われます．このカーソルを ⇨，⇦，SHIFT ↑⇦，SHIFT ↓⇨ で訂正したい個所に移動し，DEL，SHIFT INS/DEL で修正を加えていきます．

■編集用キーの使い方

①⇨，⇦ ……カーソルを左右に移動．押し続けるとリピート機能によってディスプレイの左端から，行末尾の間をカーソルが走ります．

②SHIFT ↑⇦ …シフトキーとカーソルキーの同時押しによって，カー
SHIFT ↓⇨ ソルを上下に移動することができる．

③DEL ……………カーソルのある位置の文字や記号をひとつ削除し，その間をつめる．

④SHIFT INS/DEL …カーソルのある位置と，その直前の文字や記号の間を1文字分あける．

使い方

では，次のプログラムを入力して，実際に編集をやってみましょう．

```
10 PRINT "HEIKIN"
20 INPUT A,B,C
30 H=(A+B+C)/3
40 PRINT H
50 GOTO 10
```

1 プログラムリストを最初から見たいとき

プログラムリストを最初から表示させ，1行ずつ訂正を加えていく場合は，次の使い方をします．

EDIT ↵

続いて ↵ キーを押すと，行番号の若い順に次つぎとリストを表示します．

2 行番号の若い方へ，逆にさか昇ってリストを見たいとき

先のプログラムで，先頭から50行目までをリストアップした状態で，もう一度10行目を見たいといった場合，次の操作をします．

50行目までを表示した状態

```
20  INPUT  A, B, C
30  H= (A+B+C) /3
40  PRINT  H
50  GOTO  10_
```

⇩

SHIFT ↵ を4回操作すると，右のように10行目を表示する．

```
40  PRINT  H
30  H= (A+B+C) /3
20  INPUT  A, B, C
10  PRINT "HEIKIN"_
```

3 プログラムの途中の行番号から見たいとき

プログラムの途中の行番号のリストを見たいときは，次のようにします．

EDIT　行番号 ln ↵

この行番号lnは，もちろんリストした行の番号です．以後, ↵ キーを押すごとにそれ以降の行を表示します．

4 ⇦，⇨，INS/DEL の使い方

先のプログラムの20行目と30行目を，次のように訂正します．

```
20  INPUT  A, B, C, D
30  H=(A+B+C+D)/4
```

EDIT ↵ で20行目のリストをとると，画面は次のようになります．

SHIFT EDIT(E) 20 ↵

```
Ready P0
EDIT 20
  20 INPUT A, B, C_
```

20行目は，これに続いて“，”とDを入力し ↵ キーで訂正終了です．↵ キーと同時に30行目が表示され，画面は次のようになります．

, D ↵

```
Ready P0
EDIT 20
  20 INPUT A,B,C,D
  30 H=(A+B+C)/3_
```

続いて，30行目に“+D”を挿入する操作は，

⇦ ⇦ ⇦ SHIFT INS(DEL) INS(DEL) で2つ分のスペースをあけ
同時押し

+ D ⇨ ⇨ 4 ↵

です．

■パスワードの付いたプログラムにEDITコマンドを用いると，PRエラーとなります

■エディットモードの解除

エディットモードは，次のような場合に解除されます．

①BRK キーを押したとき

②CLS キーを押したとき

③誤ってエラー操作をしたとき

④表示すべきプログラムがなくなった場合

⑤電源を，改めてOFF−ONしたとき

マニュアル・コマンド

LIST/LLIST

(list：リスト
l-list：エルーリスト)

機　能	LIST　：指定されたプログラムをディスプレイに出力する LLIST：指定されたプログラムをプリンタで打ち出す
書　式	LIST（LLIST） LIST（LLIST）ALL LIST（LLIST）V LIST（LLIST）ln LIST（LLIST）ln－ LIST（LLIST）_ln LIST（LLIST）ln_lm （1≦ln≦lm≦9999）

解　説

LIST，LLISTコマンドは，PB-700に現在記憶されているプログラムのリストを見る際に使うものです．LISTは画面に，LLISTはプリンタを通してロールペーパーに印字するという形で，それぞれ表示されます．

また，プログラムリストだけでなく，現在プログラム中で登録変数という形で使用している変数（登録済変数）も，リストアップすることができます．

使い方

LIST，LLISTコマンドには，次のような使い方があります．

① LIST（LLIST）……現在指定されているプログラムエリアのプログラムをリストアップする

② LIST（LLIST）ALL……P0～P9に入っているプログラムを全てリストアップする

③ LIST（LLIST）V……登録済変数をリストアップする

④ LIST（LLIST）行番号……現在指定されているプログラムエリアのプログラムの，指定行番号のみリストアップする

⑤ LIST (LLIST) 行番号－……現在指定されているプログラムエリアのプログラムの，指定行番号以降のプログラムをリストアップする．

⑥ LIST (LLIST) －行番号……現在指定されているプログラムエリアのプログラムの，指定行番号までのプログラムを全てリストアップする

⑦ LIST (LLIST) 行番号n－行番号m……現在指定されているプログラムエリアのプログラムの行番号nからmまでをリストアップする

では，次のリストを入力して，LIST，LLISTコマンドの使い方を見ていくことにしましょう．

```
10 PRINT "HEIKIN"
20 INPUT "A=",AB,"B=",CD,"C=",A1
30 D1=(AB+CD+A1)/3
40 PRINT D1
50 GOTO 10
```

※このリストは、LIST Vの使い方をみるために，変則的ではあるが登録変数を使用している

1 LIST (LLIST) V

これは，現在使用している登録変数や配列変数を全てリストアップするものです．LLIST V ⏎ でプリントアウトさせると，先のプログラムで使った登録変数を次のように打ち出してきます．

```
AB    CD    A1    D1
```

また，登録変数以外にDIMコマンドで宣言した配列変数についても，その配列変数名を表示します．

2 LIST (LLIST) ALL ↵

PB−700には，P0〜P9までのプログラムエリアがあり，それぞれに独立したプログラムを入力することができます．

これは，全プログラムエリアのプログラムをいちどに表示させたいときに，使います．

LLIST ALL ↵ で実行させてみると，下のようにリストをプリントアウトします．このように，行番号の小さいものから順に印字し，また，必ずプログラムエリアも印字するようになっています．

```
P0 10 PRINT "HEIKIN"
   20 INPUT "A=",AB,"B=",CD,"C=",A1
   30 D1=(AB+CD+A1)/3
   40 PRINT D1
   50 GOTO 10
```

3 LIST (LLIST) ↵

これを実行すると，現在指定されているプログラムエリアのプログラムを，行番号の小さいものから順に，全て表示(印字) します．

4 LIST (LLIST) 行番号 ↵

LIST に続いて行番号を指定すると，現在指定されているプログラムエリアの，その行番号のプログラムを表示(印字) します．例えば PROG 0 ↵ で，LIST 30 ↵ を実行すると，

```
30  D1 = (AB+CD+A1)/3
```

が表示されます．

5 LIST (LLIST) 行番号− ↵

行番号の次に−（マイナスキーで入力する）をつけると，その行番号以降のプログラムを順に表示（印字)します．例えば，LIST 30− ↵ を実行すると

```
30  D1 = (AB+CD+A1)/3
40  PRINT D1
50  GOTO 10
```

を表示します．

6 LIST（LLIST）ー行番号

行番号の前にーをつけると，その行番号までのプログラムを頭から表示（印字)します．例えば，LIST－30 ↵ で

```
10  PRINT "HEIKIN"
20  INPUT "A=", AB, "B=", CD, "C=", A1
30  D1 = (AB+CD+A1)/3
```

を表示します．

7 LIST（LLIST）行番号nー行番号m

この場合は，行番号nからmまでの間のプログラムを表示します．プログラムは，行番号の小さい方から順に表示(印字)されますから，n≦mでなければなりません．例えば，LIST 20－40 ↵ で

```
20  INPUT "A=", AB, "B=", CD, "C=", A1
30  D1 = (AB+CD+A1)/3
40  PRINT D1
```

を表示します．

テクニック

1 LIST，LIST ALLの実行を中断する

前述の LIST コマンドの使い方でプログラムリストをとると，ディスプレイには次から次へとプログラムがめまぐるしく現われ，そのためにじっくりと内容を確認するというわけにいきません．

そこで，表示を一時ストップさせるためには ↵ キーを押します．するともう一度 ↵ キーが押されるまで，表示はストップした状態になります．

なお，LISTコマンドの実行は SHIFT STOP/ANS では停止状態にはなりません．

また，途中でプログラムリストを見る必要がなくなった場合には，BRK キーを押すと LIST，LIST ALL 状態から抜け出すことができます．

2 LLIST，LLIST ALLからの抜け出し

LLIST，LLIST ALL でプリンタにプログラムを打ち出している場合には，↵ キーによる一時中断はできません．可能なのは BRK キーによってプリントアウトを中止し，LLIST 状態から抜け出すことだけです．

マニュアル・コマンド

LOAD (load：ロード)

機　能	カセットテープに記録してあるプログラムを，本体に読み込む
書　式	LOAD LOAD ALL LOAD, A LOAD, M

LOAD命令は，SAVE命令によって，カセットテープに記憶したプログラムを，再びPB−700に読み込むときに使うコマンドです．SAVEするときにつけたファイル名が役立つのは，このときです．LOADコマンドとともにファイル名を入力すると，PB−700がそのファイル名を自動的に探し出し，テープからプログラムを読み込みます．

ファイル名を指定しない場合は，最初に見つけ出したプログラムを読み込みます．

解　説

LOADコマンドの形式には，次のものがあります．

(1)LOAD………現在指定されているプログラムエリアに，SAVEまたはSAVE"ファイル名"の形式で記録されたプログラムのうち，最初に見つけ出したプログラムを読み込みます．
この場合，SAVE時のプログラムエリアと，現在指定されているプログラムエリアが異なっても，問題はありません．

(2)LOAD"ファイル名"
現在指定されているプログラムエリアに，同じファイル名でSAVEされているプログラムを探し出し，読み込む．この際，SAVE時とLOAD時のプログラムエリアは異なっても良い．

(3)LOAD ALL…SAVE ALL形式で記録されたプログラムを，それぞれ同じプログラムエリアに読み込む．ファイル名を付けないので，SAVE ALL形式で記録されたプログラムのうち，最初に見つけ出したものを読み込みます．

(4)LOAD ALL **"ファイル名"**

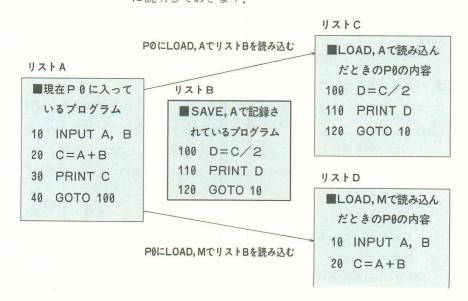
SAVE ALL"ファイル名" で記録したプログラムのうち，同じファイル名のものを探し出し，PB−700に読み込む．

(5)LOAD，A／LOAD **"ファイル名"**，A

アスキーコード方式（SAVE, AまたはSAVE"ファイル名"，A）で記録されたプログラムを，PB−700に読み込みます．ファイル名がない場合は，最初に見つけ出したプログラムを，またファイル名がある場合には，同じファイル名のものを探し出して読み込む．

(6)LOAD，M／LOAD **"ファイル名"**，M

SAVE, A または SAVE **"ファイル名"**，A で記録されたプログラムを読み込むものです．

同様のものにLOAD, A　LOAD **"ファイル名"**，A がありますが，これとの違いは，行番号さえ異なれば，古いプログラムも消えずに残るということです．これについて以下に説明しておきます．

●通常，LOADすると，同時にNEWの機能も働き，前のプログラムはそっくり消えてしまう．しかしLOAD,Mを使うと，行番号さえ異なれば，前のプログラムも消えずに残っている．

```
30  PRINT C
40  GOTO 100
100 D=C/2
110 PRINT D
120 GOTO 10
```

使い方

1 SAVE時の形式と同じ形式のもの以外LOADしない

SAVEで記録したものはLOADで，SAVE ALLで記録したものはLOAD ALLでというように，同じ形式で対応しています．以下，LOAD可能なものの組み合せを示します．

	LOAD	LOAD "ファイル名"	LOAD ALL	LOAD ALL "ファイル名"	LOAD A	LOAD "ファイル名", A	LOAD, M	LOAD "ファイル名", M
SAVE	○	×	×	×	×	×	×	×
SAVE "ファイル名"	○	○	×	×	×	×	×	×
SAVE ALL	×	×	○	×	×	×	×	×
SAVE ALL "ファイル名"	×	×	○	○	×	×	×	×
SAVE, A	×	×	×	×	○	×	○	×
SAVE "ファイル名", A	×	×	×	×	○	○	○	○

2 パスワードについて

パスワードのついたプログラムをSAVEすると，パスワードそのものも同時に記録されます．このようなプログラムをLOADする場合についてまとめます．

①PB−700本体にパスワードがついていない場合は，LOADが可能．この場合，LOADしたパスワードが，そのままPB−700のパスワードとして生きてきます．

②PB−700本体に，LOADするプログラムについているパスワードと同じものがついている場合，そのままLOAD可能．この場合にもパスワードは生きています．

③PB−700本体のパスワードと，LOADするプログラムのパスワードが異

なる場合には，読み込み開始と同時にPRエラーを表示して，カセットテープが止まってしまいます.

3 LOAD中のエラーについて

①RWエラー

これは，LOAD中にパリティーエラーが発生したときに起こるものですが，この場合にはNEW ⏎ でそれまでにLOADしたプログラムを消し，再度LOADし直します.

②OMエラー

これは，メモリー容量が不足したときに発生します．この場合には，不要なプログラムを消すか，RAMを増設しなければなりません.

参照　SAVE　VERIFY　CHAIN　PUT　GET

マニュアル・コマンド

NEW/NEW ALL (new　：ニュー / new all：ニューオール)

機　能	プログラムの消去
書　式	NEW NEW ALL

新しくプログラムをコンピュータに入力する際は，前もって先に入っているプログラムを消しておく必要があります．このようにプログラムを消去するためのコマンドがNEW/NEW ALLです．

解　説

1 NEWコマンドの機能

NEWコマンドは，現在指定されているプログラムエリアのプログラムを消去します．ですから，消したいプログラムのエリアをPROGコマンドで指定し，NEWを実行します．NEWコマンドは，パスワード（PASSの項148ページ参照）のついている状態で実行するとPRエラーを表示し，受けつけてくれません．

また，変数のクリアを行なうことはできません．（CLEARの項160ページ参照）

2 NEW ALLコマンドの機能

NEW ALLコマンドは，全プログラムエリアのプログラムを，一度で消去してしまうものです．このコマンドは，PASSコマンドの実行状態でも有効ですので，使用するときは十分な確認が必要です．

また，プログラムだけでなく，変数も全てクリアし，初期状態に戻してしまいます．つまり

①数値変数を全てクリアして0にする

②文字変数を全てクリアして〝〃（ヌルストリングス）にする

③登録変数は全て消去される

④DIM命令（DIMの項 162ページ参照）による配列宣言を解除する

さらに，プログラムエリアを0に設定し直し，ANGLEも0，つまりDEG（デグリー）に設定される．

このように，NEW ALLは全てのRAM内容を，初期の状態に設定し直すという働きがあります．

使い方

NEW，NEW ALLとも，プログラム中で使うことはできません．使用する場合は，直接キー入力で

NEW ⏎ または NEW ALL ⏎

とします．

マニュアル・コマンド

PASS　(pass：パス)

機　能	プログラムにパスワードをつけてプログラムを保護する
書　式	PASS "パスワード"

解説

やっとの思いで作り上げたプログラムも，誤って消してしまったり，他のプログラムを書き込んでダメにしてしまう，といったことが起りがちです．また他の人が勝手に消してしまうといったことも，ままあることです．

そこで，大切なプログラムに鍵をかけて，修正や消去ができないようにするのがこのコマンドです．

使い方

1 PASSコマンドの使い方と機能

PASSコマンドは，次のようにして使います．なお，プログラム中では使えません．

PASS "8文字以内のパスワード" ↵

これで，プログラムの修正，消去はPASSコマンドを解除しない限りいっさいできなくなります．実際には次のコマンドが実行できなくなります．

①DELETE
②EDIT
③LIST，LLIST
④NEW

新たにプログラムを書き込むことも不可能です．このように，PASS コマンドは，全プログラムエリアに有効に働きます．逆の言い方をすれば，特定のプログラムエリアに限って，PASSコマンドを働かせることはできません．

PASS 命令が入っているときに，上記のコマンドを実行すると，PR エラーを表示してそのことを知らせてくれます．

2 PASSコマンドの解除

パスワードは，８文字以内の文字，数字，記号を使うことができます．これを解除するには，もう一度，全く同じパスワードを使って

PASS “8文字以内のパスワード” ⏎

を入力すれば解除されます．ですから，最初に入力したパスワードを忘れてしまうと，永久に PASS 命令を解除することはできないことになります．もちろん，パスワードを見ることもできません．そこでパスワードは，自分の名前など絶対に忘れないものを使うようにします．また入力するときに十分確認することも大切です．

万一，パスワードを忘れてしまったり，自分が考えているパスワードが通用しないような場合は，やむを得ない手段ですが，電池を外すかまたは NEW ALL コマンドを実行する以外に方法はありません．しかしその場合は，全プログラムエリアのプログラムが消えてしまいますので，SAVE コマンドによってカセットテープに記録しておくことも必要になってきます．

参照　SAVE，LOAD，NEW

マニュアル・コマンド

PROG

(program：プログラム)

機　能	使用したいプログラムエリアを指定する
書　式	PROG 数値（または数式）　0≦数値(数式)<10

解　説

PB－700には，独立したプログラムを書き込むことができる，P0～P9までの10個のプログラムエリアがあります．

このプログラムエリアを指定する場合に，このPROGコマンドを使います．

使い方

まず，PB－700の電源をONにしてみて下さい．するとディスプレイには次のように表示されます．

電源ON

```
Ready P0
_
```

このように，電源をONにすると，プログラムエリアは自動的にＰ０に指定されます．では続いて，プログラムエリアＰ９を指定してみましょう．

電源ON

```
Ready P0
PROG 9
Ready P9
_
```

以上で指定は終了で，プログラムエリアＰ９が入力待ちの状態となりました．

なお，PROGに続いて数式を入れてもかまいませんが，計算結果（Ｘとすると）が０≦Ｘ<10の値をとるようにしなければなりません．この範囲を越える場合は，BSエラーを表示します．

（例）　PROG (100/20) ⟶PROG 5でP5を指定
　　　PROG (100/10) ⟶BSエラーを表示
　　　PROG (100/15) ⟶P6を指定（小数部は切り捨てられる）

参　照　GOTO，GOSUB

マニュアル・コマンド

RUN (run：ラン)

機　能	プログラムを実行する
書　式	RUN RUN行番号　(0≦行番号≦9999)

現在指定されているプログラムエリアのプログラムを，実行するためのコマンドがRUNです．

解　説

RUNコマンドは，次の2通りの使い方ができます．

①RUN ↵ ………現在指定されているプログラムエリアの，先頭の行番号から実行を開始します．実行は，行番号の小さい順に行なわれます．

②RUN 行番号 ↵ …RUNの後の行番号から実行を開始します．指定した行番号がない場合は，その行番号に最も近く，かつ指定行番号より大きな行から実行をスタートします．

なお，各プログラムエリアのプログラムの，先頭から実行を始めたい場合には，RUNコマンドを使わないで行なう方法があります．それは

SHIFT　P0 [0] ～ P9 [9]

というやり方です．つまり，SHIFTキーと[0]～[9]のテンキーとの同時押しにより，P0～P9のプログラムを実行させることができるのです．

マニュアル・コマンド

SAVE　（save：セーブ）

機　能	プログラムをカセットテープに記録する
書　式	SAVE SAVE ALL SAVE, A SAVE "ファイル名" SAVE ALL "ファイル名" SAVE "ファイル名", A

解　説

苦労して作ったプログラムは，カセットテープにとって保存し，自分のプログラムライブラリーを作るのもまた楽しいことです．

SAVEコマンドは，カセットテープにプログラムを記録するときに使うものです．このコマンドの形式には次のものがあります．

(1)SAVE　………現在指定されているプログラムエリアのプログラムを，バイナリーコード（内部コード）形式で記録する．

(2)SAVE ALL …全てのプログラムエリアのプログラムを，バイナリーコード形式で記録する．

(3)SAVE, A　……現在指定されているプログラムエリアのプログラムを，アスキーコード形式で記録する．

●アスキーコード形式でのSAVEは，バイナリーコード形式に比べ時間も長く，カセットテープの消費量も多くなるのですが，LOAD, M（LOADコマンドの項参照）を実行させるためには，このアスキーコード形式でのセーブが必要になります．

使い方

実際にカセットテープにプログラムをセーブする場合には，それぞれ「ファイル名」をつけておきます．というのは複数のプログラムを整理したり，再びカセットテープからPB−700にプログラムを読み込むとき，大変便利だからです．

1 ファイル名は，次のようにつける．

SAVE "ファイル名"

SAVE ALL "ファイル名"

SAVE "ファイル名", A

- このファイル名には，どんな文字，記号を使ってもかまいませんが，必ず8文字以下でなければなりません．
- パスワードがついたプログラムは，パスワード自身も同時にデータとして出力されます．

2 テープカウンタの数を記録する

カセットテープレコーダーを録音状態にして ↵ キーを押すと録音が開始され，終了するとリモート機能によりテープは自動的に止まります．ここで，録音開始と終了をテープカウンタの数で記録しておいて下さい．次のプログラムを読み込む際に，大変便利です．

3 正しく記録されたかどうかの確認

カセットテープへの録音が正しく行なわれたかどうかを，VERIFYコマンド（VERIFYの項155ページ参照）を使って確認して下さい．正しく記録されていなければ，再度録音し直して下さい．

☞ ファイルの属性

SAVEまたはPUTしたプログラムデータは，LOAD,GET で本体に読み込むとき，そのファイルの属性名を表示します．この属性は，SAVEやPUTのやり方で決められます．

出力時の操作	入力時の表示	出力時の操作	入力時の表示
SAVE	PF B	SAVE ALL "ABC"	ABC AF B
SAVE ALL	AF B	SAVE "ABC", A	ABC PF A
SAVE, A	PF A	PUT A	DF A
SAVE "ABC"	ABC PF B	PUT "ABC" A	ABC DF A

このように，ファイル名はそのまま表示されますが，以下，ファイルの属性を示す部分について説明しておきます．

〔入力時の表示〕

PB-700　PF　B

ファイル名　ⒶⒷ　Ⓒ

Ⓐ P：Program
　A：All program
　D：Data

Ⓑ File

Ⓒ B：Binary（内部コード形式）
　A：Ascii（アスキー）

マニュアル・コマンド

SYSTEM (system：システム)

機　能	プログラムエリアの使用状態，残りByte数，設定アングルを表示する
書　式	SYSTEM

解　説

SYSTEMコマンドは，PB−700のプログラムエリアの使用状態を表示させるためのものもです．

まず，SYSTEM ⏎ と入力してみて下さい．もし，PB−700にプログラムが何も入っていなければ，ディスプレイには次のように表示されるはずです．

```
P 0123456789
2864 Bytes：ANGLE 0
Ready P0
_
```

このような表示を出すために使うのが，SYSTEM コマンドです．ではこの表示は，何を意味しているのでしょうか．

(1) 1行目のP0123456789は，P0〜P9のプログラムエリアに，プログラムが書き込まれているかどうかを表示しています．

例えば，10 PRINT "CASIO PB−700" ⏎と入力してみて下さい．

```
P ♥123456789
2846 Bytes：ANGLE 0
Ready P0
_
```

となります．つまり，プログラムが書き込まれている場合，♥でそのことを知らせます．

(2) 2行目の2846Bytesは，使用可能な残りバイト数の表示です．なお，この数値はRAMパックの増設により異なります．

(3) 2行目のANGLE 0は，設定角度モードです．電源をONした直後には，ANGLE 0でDEG（デグリー）に設定されます．

(4) 3行目のReady P0は，プログラムエリアP0に入力可能な状態を示しています．電源ON直後はいつでもP0に設定されます．

マニュアル・コマンド

VERIFY (verify：ベリファイ)

機　能	カセットテープにSAVEしたプログラムやPUTしたデータのパリティチェックを行なう
書　式	VERIFY VERIFY "ファイル名"

解　説

SAVE命令によってカセットテープに記録されたプログラムやPUT命令により記録されたデータが，正しく記録されているかどうかを調べるのがVERIFYコマンドです．

■このコマンドの形式には，次の２通りがあります．

① VERIFY ……カセットテープを回しながら，最初に見つけたプログラムをチェックする

② VERIFY "ファイル名"
……カセットテープを回しながら，同じファイル名のプログラムを探し出し，チェックを行なう

■ SAVE コマンドの６つの形式（SAVE の項参照）で記録されたプログラムについて，全て VERIFY ⏎ または VERIFY "ファイル名" ⏎ でチェックを行ないます．つまり，SAVE の形式を指定する必要はありません．

■SAVEが正しく行なわれていない場合は，RW エラーを表示します．その場合は，もう一度 SAVE をしなければなりません．

使い方

次に，実際の手順をまとめておきます．

手順1：テープの巻き戻し－SAVE コマンドで記録したテープを，カウンタを利用して最初の位置に戻す．

手順2：VERIFY 命令を入力する－ファイル名があれば VERIFY "ファイル名" ⏎ ，なければ VERIFY ⏎

手順3：カセットテープレコーダのPLAYボタンを押す．チェックが始まったらPF B, AF B, PF A（153ページ参照）のいずれかを表示する．正しく SAVE されていれば，Ready P0～P9を表示してカセットも止まる．誤りがあればRWエラーを表示して，カセットも止まる．

プログラム・コマンド

ANGLE　(angle：アングル)

機　能	角度単位の設定
書　式	ANGLE 数式 (0≦数式＜3)

日常，角度の単位は30°，60°などの使い方をしますが，これはDEGREE(ディグリー)と呼ばれる角度の単位です．数学では，これ以外にRADIAN(ラジアン)，GRADIENT(グラディエント)の使用頻度が多くなりますが，PB—700はそのいずれにも対応できます．

解　説

ANGLEコマンドは，次の3つの角度単位を指定・変更するときに使うものです．

①DEGREE ……(例)45°，90°　〈入力範囲x〉 −5400 ＜x＜5400

②RADIAN ……(例)0.5π，2π〈入力範囲x〉 −30π ＜x＜30π

③GRADIENT….(例)300，1000〈入力範囲x〉 −6000 ＜x＜6000

以上3つの角度単位の関係は，次の通りです．

360DEG (＝360°) ＝2πRAD＝400GRA

ANGLEコマンドを使った角度単位の指定方法は，以下の通りです．

ANGLE 0 ⟶DEGREEを指定

ANGLE 1 ⟶RADIANを指定

ANGLE 2 ⟶GRADIENTを指定

電源をONにした直後はANGLE 0の状態，つまりDEGREEに指定されています．

サンプル・プログラム

```
10 REM *** ANGLE ***
20 ANGLE 0:PRINT SIN30;
30 ANGLE 1:PRINT SIN(PI/6);
40 ANGLE 2:PRINT SIN(100/3)
50 END
```

このプログラムは，3つの角度単位でSINの値を表示させるプログラムです．
全て0.5となります．

プログラム・コマンド

BEEP　(beep：ビープ)

機　能	ブザー音を発生する
書　式	BEEP BEEP 0 BEEP 1

PB−700には,ブザー機能がついています．このブザーを鳴らす命令がBEEPです．

ブザー音の利用法はいろいろありますが，例えば実行に時間がかかるプログラムでは，実行の終りの部分にBEEP命令を入れておけば，ブザーで実行終了を知らせてくれます．また，ゲームではこれを使うことにより，楽しさが倍増します．

解　説

BEEPコマンドの形は，次の3通りです．

①BEEP ……比較的低いブザー音を発生する

②BEEP 0…BEEPと同じ音を発生する

③BEEP 1…やや高いブザー音を発生する

サンプル・プログラム

```
100 REM *** BEEP ***
110 IF INKEY$="" THEN 110
120 IF INKEY$="0" THEN BEEP 0
130 IF INKEY$="1" THEN BEEP 1
140 GOTO 110
```

このプログラムは，0を押すと低いブザー音が，1を押すと高い方のブザー音を鳴らせるプログラムです．

CHAIN

(chain：チェイン)

機　能	指定プログラムをLOADして，先頭行番号より実行する
書　式	CHAIN CHAIN“ファイル名”

解　説

プログラム実行中にCHAIN命令が出てきたときは，そこでプログラムの実行を中止し，CHAIN命令で指定されたファイル名のプログラムを，マイクロカセットからLOADして，その先頭から実行します．

ファイル名がない場合は，SAVEまたはSAVE“ファイル名”で記録されたプログラムのうち，最初に見つけ出したものをLOADします．

CHAINコマンドの形は，次の2通りがあります．

①CHAIN ………最初に見つけ出したPF B（SAVE，SAVE“ファイル名”で記録したプログラム）をLOADし実行する．

②CHAIN“ファイル名”

………指定されたファイル名のPF BをLOADして実行する．

■現在指定されたプログラムエリアにLOADしますので，現在入っているプログラムはNEWされてしまう．

■SAVE ALL，SAVE, Aで記憶されたプログラムは，CHAINコマンドでは読み込みはできない．

■LOADするプログラムにパスワードがついている場合は，そのパスワードも読み込む．

■CHAINが実行されても，変数はクリアされない．

使い方

リスト1～3を，プログラムエリアP1～P3に入力し，カセットテープに記録して下さい．

〈リスト1〉……円の面積を計算するプログラム

〈リスト2〉……三角形の面積を計算するプログラム

〈リスト3〉……四角形の面積を計算するプログラム

さらに，〈リスト0〉をP0に入力して実行して下さい．〈リスト0〉は，70～90行目のCHAINコマンドで，〈リスト1〉～〈リスト3〉のそれぞれを選択し，PB−700に読み込み，計算を実行します．

リスト0

```
10 REM CHAIN PRO.0
20 CLS :PRINT "MENSEKI KEISAN"
30 PRINT "1";CHR$(180);CHR$(221);"  2
   ";CHR$(187);CHR$(221);CHR$(182);CHR$(184
   );
40 PRINT "  3";CHR$(188);CHR$(182);CH
   R$(184)
50 PRINT "SELECT NO.    "
60 BB$=INKEY$:IF VAL(BB$)>3 THEN 60 E
   LSE IF VAL(BB$)<1 THEN 60
70 IF BB$="1" THEN CHAIN "PRO.1"
80 IF BB$="2" THEN CHAIN "PRO.2"
90 CHAIN "PRO.3"
```

リスト1

```
10 REM PRO.1
20 PRINT "HANKEI";
30 INPUT RR
40 S=PI*RR^2
50 PRINT S
60 END
```

リスト2

```
10 REM PRO.2
20 PRINT "TAKASA";
30 INPUT HH
40 PRINT "TEIHEN";
50 INPUT LL
60 S=HH*LL/2
70 PRINT S
80 END
```

リスト3

```
10 REM PRO.3
20 PRINT "TATE";
30 INPUT HH
40 PRINT "YOKO";
50 INPUT LL
60 S=HH*LL
70 PRINT S
80 END
```

プログラム・コマンド

CLEAR (clear：クリアー)

機　能	全ての変数をクリアする
書　式	CLEAR

解　説

CLEAR コマンドは，全ての数値変数，文字変数をクリアするための命令です．

数値変数は0に，文字変数は“ ”（ヌルストリング）になります．また同時に，プログラム中で登録した登録変数は，その登録が抹消されますし，定義された配列変数も，その定義を解除してしまいます．

さらに，FORネスティングのスタックもクリアされますので，FOR～NEXTループを続けることができなくなります．

使い方

下の左のプログラムはデータの総和とデータの個数を集計して表示するものです．しかし，いったんBRKキーを押して，もう一度このプログラムを実行させると，変数S，Nに1回目の数値がそのまま入っているのでうまくいきません．そこで，10行目と30行目の間にCLEARコマンドを入れて右のようにしました．これで，何回実行しても，その都度正しい答が得られます．

```
10 PRINT "TOTAL"
30 INPUT "D=",D
40 S=S+D
50 N=N+1
60 PRINT "S(";N;")=";S
70 GOTO 30
```

```
10 PRINT "TOTAL"
20 CLEAR
30 INPUT "D=",D
40 S=S+D
50 N=N+1
60 PRINT "S(";N;")=";S
70 GOTO 30
```

プログラム・コマンド

CLS

(clear screen：クリアスクリーン)

機　能	表示を全てクリアし，ホーム位置（左上隅）にカーソルを移す
書　式	CLS

解　説

CLSは，クリアスクリーンの略で，画面を消去してカーソルを画面左上隅のホームポジションに移動するものです．

これは，グラフィック表示などをする場合，最初にいったん画面をクリアしておくようなときに使われます．

①マニュアルで実行した場合は，カーソルは（0，1）の位置に表示されます．

②プログラム中での実行時には，カーソルは（0，0）の位置に設定されます．

使い方

```
10 CLS
20 FOR I=0 TO 360 STEP 12
30 DRAW(SINI*15+80,COSI*15+15)
40 NEXT I
50 END
```

このプログラムは，次のような図形を表示するものです．

このように，画面に図形を描く場合には，描き始める前に画面をクリアしておかなければなりません．そのために，プログラムの先頭でCLSコマンドを使っています．

プログラム・コマンド

DIM

(dimension：ディメンション)

機　能	配列を宣言する
書　式	DIM 配列変数名(添字)〔,配列変数名(添字)〕

DIM 命令は，メモリーエリアに指定する変数名で，配列を作ることを宣言します．DIMで宣言された変数は，配列変数と呼ばれ，単精度数値配列，半精度数値配列，文字配列の3種類からなります．

解説

DIMの宣言のしかたは，次のように示すことができます．

DIM 配列変数名（添字〔,添字〕）〔,配列変数名（……

（添字の最大値は，255までです）

以下に配列の種類による宣言文の例を示します．

書　式	種　類	例
DIM配列変数名(I)	1次元単精度数値配列	DIM A(5)
DIM配列変数名(I, J)	2次元単精度数値配列	DIM A(2,3)
DIM配列変数名！(I)	1次元半精度数値配列	DIM A!(5)
DIM配列変数名！(I, J)	2次元半精度数値配列	DIM A!(2,3)
DIM配列変数名$(I)*N	1次元文字配列	DIM A$(5)*20
DIM配列変数名$(I, J)*N *Nは省略できます	2次元文字配列	DIM A$(2,3)*20

ここで，I，J，Nは，0≦I＜256，0≦J＜256，0≦N＜80の実数，あるいは数式で，数値の小数点以下は切り捨てられます。

使用可能な配列変数名は英大文字のA～Zで，1文字のみです．

また，次元の最大値は2で，1次元配列，2次元配列を設定できます．

配列変数名の直後に“！”をつけることによって半精度数値配列を，“$”をつけることによって文字配列を設定できます．

また，文字配列では，“＊N”をつけることによって，“N”の長さをもつ文字配列が代入できる文字配列を宣言します．ただし，“＊N”を省略したときは，16文字の文字配列となります．

使い方

次のプログラムを入力して，RUNしてみてください．

```
 10  CLEAR
 20  DIM A(2,3), B(2,3)
100  END
```

20行～100行の間に，何か演算式を書いてもよいのですが，何も書き込まなくてもかまいません．

実行後，Ready P0などの文字が出たら，次のように，配列変数のリストを出してみます．

```
LIST   V ↵
A( )   B( )
Ready  P0
```

……のように表示されます．このように，DIMで宣言した配列変数名は……

```
LIST  V ↵
```

……を用いることによって，知ることができます．

では，上記のプログラムに，次のリストを追加して，各配列変数の内容を見てみましょう．配列変数を使う場合は，必ず，その前にDIMで宣言しておかねばなりません．

```
30  FOR I=0 TO 2
40  FOR J=0 TO 3
50  PRINT A(I,J);B(I,J);
60  NEXT J:NEXT I
```

30～60行を追加して、RUNすると，次のように，表示されます．

■**実行例**

これは，A(0,0), B(0,0)～A(2,3), B(2, 3) の24個の配列変数の内容を表示したわけですが，すべて,“0” となっています．

ここで重要なことは，DIM 命令が実行されると，その配列の内容はすべて“0” となるということです．

ただし，上記のような数値配列は “0” を表示しますが，文字配列を宣言した場合は，すべてヌルストリングとなり，何も表示されません．ヌルストリングは，文字配列が空っぽであることを意味します。

ここで注意しなければならないことは，スペースとヌルストリングの違いです．スペースは空白が入っているもの(A$(I)=“␣”) で，ヌルストリングは，何も入っていない状態 (A$(I)=“ ”)ですので注意して下さい．

サンプル・プログラム1

●**並べかえ（ソート）プログラム（1次元数値配列）**

```
10 CLEAR
20 DIM D(5)
30 FOR I=1 TO 5
40 PRINT "DATA";I;" =";:INPUT D(I)
50 NEXT I
60 REM NARABIKAE
```

```
 70 F=0
 80 FOR I=1 TO 4
 90 IF D(I)<D(I+1) THEN X=D(I):D(I)
    =D(I+1):D(I+1)=X:F=1
100 NEXT I
110 IF F=1 THEN 70
120 REM KEKKA
130 FOR I=1 TO 5
140 PRINT D(I);
150 NEXT I
160 END
```

このプログラムは，５つの数値データを入力し，そのデータを大きい順に並べかえるプログラムです。

20行の DIM 命令で，D(0)～D(5)の配列の使用を宣言していますが，ここでは，D(1)～D(5)の５つの配列を使っています．

60行～110行が並べかえのプログラム，120行～150行が，大きい順に並べかえた結果を表示するプログラムです．

このサンプル・プログラムでもわかりますように，配列変数は FOR～NEXT 命令と組合せて使うと便利です．

サンプル・プログラム２

●タテ・ヨコ集計（２次元数値配列）

```
10 DIM A(3,3),X(3),Y(3)
20 FOR I=1 TO 3
30 FOR J=1 TO 3
40 PRINT "(";I;",";J;")";"="
```

```
 50 INPUT A(I,J)
 60 NEXT J:NEXT I
 70 REM SHOUKEI
 80 FOR I=1 TO 3
 90 FOR J=1 TO 3
100 X(I)=X(I)+A(I,J)
110 Y(J)=Y(J)+A(I,J)
120 NEXT J:NEXT I
130 REM KEKKA
140 FOR I=1 TO 3
150 PRINT "X(";I;")=";X(I)
160 PRINT "Y(";I;")=";Y(I)
170 FOR K=1 TO 500:NEXT K
180 NEXT I
190 END
```

このプログラムは，表1のようなデータを，表2のような2次元配列に割り当て，タテ・ヨコの小計X(1), X(2), X(3)およびY(1), Y(2), Y(3)を求めるプログラムです．

14	9	21	X(1)
35	4	53	X(2)
6	15	11	X(3)
Y(1)	Y(2)	Y(3)	

A(1,1)	A(1,2)	A(1,3)	X(1)
A(2,1)	A(2,2)	A(2,3)	X(2)
A(3,1)	A(3,2)	A(3,3)	X(3)
Y(1)	Y(2)	Y(3)	

80行～120行が小計を求めるプログラム，130行～180行が求めた値を表示するプログラムです．

参照　CLEAR, NEW　ALL, ERASE, LIST V

プログラム・コマンド

DRAW／DRAWC （draw：ドゥロー　drawclear：ドゥロークリア）

機　　能	DRAW：ドット(点)を描く DRAWC：ドット(点)を消す
書　　式	DRAW(X_1, Y_1)〔−(X_2, Y_2)〕 DRAWC(X_1, Y_1)〔−(X_2, Y_2)〕

DRAW 命令は，スクリーン上にドット(点)，あるいは線を描くもので，DRAWC は逆に消すものです．

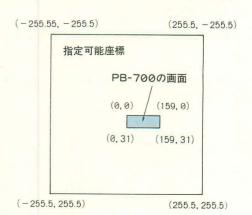
画面上に，キャラクタのみならず，点，線を表わすことができるので，このコマンドは，画面にグラフ等を描くことができます．

解　説

DRAW(X, Y)，または，DRAWC(X, Y)で指定可能な座標(X, Y)は……．

$-255.5 < X < 255.5$　　　$-255.5 < Y < 255.5$

……となっています．

画面のドットの座標は，$0 \leqq X \leqq 159$，$0 \leqq Y \leqq 31$となっていますので，次の図のように画面を表示します．

図の□の部分が画面に相当します．

ただし，画面の左上隅が，原点（0, 0）となっています．

使い方

DRAW, DRAWC の使い方を，次にまとめておきます．

DRAW (X1, Y1)	座標点(X1, Y1)に点を描く
DRAWC(X1, Y1)	座標点(X1, Y1)の点を消す
DRAW (X1, Y1)−(X2, Y2)	座標点(X1, Y1)から座標点(X2,Y2)に線を描く
DRAWC(X1, Y1)−(X2, Y2)	座標点(X1, Y1)から座標点(X2,Y2)の線を消す

上記のX，Yは数値，変数名，および数式であり，その値のとり得る範囲Xは，−255.5＜X＜255.5，−255.5＜Y＜255.5

……です．実際のスクリーン上では，小数点以下第一位目を四捨五入した整数値で示されます．

次のプログラムは，画面上に，長方形を描きます．

```
10  REM DRAW
20  CLS
30  DRAW(10,10)-(10,20)-(150,20)-(150,10)-(10,10)
40  END
```

このプログラムのように，"−" で座標をいくつも連続させると，図が描けます．

DRAW で直線を引く時，演算誤差により画面の縁の点を描かない場合があります．

この場合は該当する点についてのみ点を描くようにするか，または誤差を補正するために直線の終点（x_2，y_2）が画面の縁（$x_2=159$ もしくは $y_2=31$）となるようした上で，終点を（x_2，y_2+0）として下さい．

DRAW(x_1, y_1)−(x_2, y_2+0)

$x_2=159$ もしくは $y_2=31$

サンプル・プログラム

●キャラクタを2倍に拡大して表示するプログラム

```
10 CLEAR :DIM A(7,7):CLS
20 K$=INKEY$
30 IF K$="" THEN 20
```

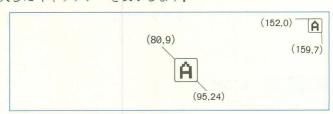
```
 40 LOCATE 19,0:PRINT K$
 50 FOR I=0 TO 7
 60 FOR J=0 TO 7
 70 A(I,J)=POINT(I+152,J)
 80 NEXT J:NEXT I
 90 FOR I=0 TO 7
100 FOR J=0 TO 7
110 IF A(I,J)<>1 THEN 160
120 DRAW(2*I+80,2*J+9)
130 DRAW(2*I+81,2*J+9)
140 DRAW(2*I+80,2*J+10)
150 DRAW(2*I+81,2*J+10)
160 NEXT J:NEXT I
170 LOCATE 0,1
180 END
```

このプログラムは，INKEY$ でキャラクターを１文字入力し，その文字を，下図のような，LOCATE (19,0)の位置にまず表示させます．

この文字は，(152.0)－(159.7)を対角線とする正方形の位置にドットで表示されますが，POINT 関数で点灯しているドットと消えているドットをチェックし，その値０，１をＡ(Ｉ，Ｊ)の配列変数に代入します．

この配列変数をデータとして，座標(80,9)－(95,24)を対角数とする正方形の位置へ，２倍に拡大したキャラクターを表示します．

参　照　POINT

プログラム・コマンド

END　(end：エンド)

機　能	プログラムの実行を終了させる
書　式	END

END命令は，プログラムの実行を終了させます（END命令が実行されると，今までのネスティング・スタックをクリア，つまりFOR～NEXTループやGOSUBの途中でプログラムを終了させます）．

END文は，プログラムのどこに置くこともできますし，いくつ置いてもかまいません．

また，プログラムの最後のEND命令は省略することができます．

サンプル・プログラム

```
500  FOR I=0 TO 1000
510  K$=INKEY$
520  IF K$="A" THEN 1000
530  IF K$="B" THEN 2000
540  IF K$="C" THEN 3000
550  IF K$="D" THEN END
560  NEXT I
570  END
```

このプログラムは，メニュープログラムの一部として使われるものです．

"A"，"B"，"C"などの文字が入力されると，それぞれ1000行，2000行，3000行のプログラムへ飛び，それらのプログラムを実行します．

"D"が入力されれば，即座に終了しますが，"A"～"D"以外のキーが入力されたり，キーが押されないと，ある程度，時間を置いて自動的に終了します．

参照　STOP

プログラム・コマンド

ERASE (erase：イレース)

機　能	登録変数，配列変数を，1変数名単位で解除する
書　式	ERASE　変数名〔,変数名〕

ERASE 命令は，LIST V によって確認できる登録変数や配列変数を，1変数名単位で解除することができます．

解除の対象となる変数名の指定方法は，次のように行ないます．

LIST V により表示される変数名　AB，AB$，A(　)，A!(　)，A$(　)

ERASE　AB, AB$, A, A!, A$

このERASE命令の実行により，FORネスティングスタックはクリアされます．

登録されていない変数名を指定した場合は，何もしないで，次の実行に移ります．

参 照　DIM，LIST V

プログラム・コマンド

FOR～TO～STEP/NEXT (for～to～step/next：フォー・トゥ・ステップ/ネクスト)

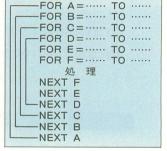
機　　能	FORからNEXTまでを指定回数繰り返す
書　　式	FOR 数値変数＝i TO j〔STEP k〕 ∫ NEXT 数値変数（FOR 文と同一変数） i：変数の初期値　j：変数の終値　k：増分

FOR～NEXT命令は，FOR命令とNEXT命令にはさまれた各ステートメントを，終値を越えるまで繰り返し実行します．

解　説

この命令は，上記の書式にもとずいて書きますが，いくつかと制約がありますので，次に示しておきます．

(1) 変数は数値変数でなくてはならない．

(2) STEP kは省略することができるが，その場合には増分として＋1（STEP 1）が設定される．

(3) 初期値i，終値j，増分kには変数や数式を用いることができる．

(4) k＞0 でかつ i＞j の場合は，FOR～NEXT間は1回だけ実行される．

(5) k＜0 でかつ i＜j の場合も，FOR～NEXT間は1回だけ実行される．

(6) FOR～NEXT間を実行して，次の行に移った場合の制御変数は，jより大きなj＋nkの値となる．

(7) FOR～NEXTのネスティングは6段まで可能．

(8) FOR～NEXTループは完全に入れ子となっていなくてはならず，区間が交錯している場合はFOエラーとなる．

(9) 1つのFORに対するNEXTはいくつ存在してもよい．

(10) FORが実行されないで，NEXTが出現した場合はFOエラーとなる．

(11) NEXT命令の変数は省略できない．

(12) ERASE，CLEARが実行されると，FORネスティングのスタックがクリアされるため，次のNEXT文でFOエラーとなる．

(13) 同一制御変数を用いた場合には，新しいループが有効となる．
ただし，新しいループの内側にあるループはすべてクリアされる．

(14) FOR〜NEXTループ外から，GOTO，GOSUBなどでループ内に飛び込むことはできない．

使い方

次のプログラムは，初期値，終値，増分の値によって，変数がどのように変化するかを見るプログラムです．

```
10  INPUT "FOR I=",I,"TO",J,"STEP",K
20  PRINT "FOR I=";I;"TO";J;"STEP";K
30  FOR A=I TO J STEP K
40  PRINT "HENSUU:A=";A
50  FOR X=1 TO 100:NEXT X
60  NEXT A
70  PRINT:GOTO 10
```

いくつかの実行例を次に示しておきます．

次のプログラムは，FOR〜NEXTループを2段重ねたものです．

```
10  CLEAR
20  DIM A(9,9)
30  FOR I=1 TO 9
40  FOR J=1 TO 9
50  A(I,J)=I*J
60  NEXT J
70  NEXT I
```

この例では，2段のFOR〜NEXT間で，配列変数A(I，J)に九九を演算し，代入しています．

内側の FOR～NEXT 命令で9回，外側の FOR～NEXT 命令で9回実行しますので，合計9×9回の演算を行なっています．

このプログラムでは，FOR～NEXT 命令を2段重ねていますが，FOR と NEXTは，内側は内側どうし，外側は外側どうしで対応していなければなりません．これは，3段～6段においても同様です．

次のプログラムは，FOR～NEXT命令の演算の結果によって，ループから飛び出し，流れを変えるものです．

```
10  CLEAR
20  FOR I=1 TO 100
30  X=X+I
40  IF X>=1000 THEN 100
50  NEXT I
60  END
100 PRINT I;X
110 END
```

このプログラムは，1＋2＋3……のたし算をして，その過程で，最初に1000を超える結果が出たとき，100行へ飛び出し，そのときの変数の値とたし算の結果を表示するものです．

このように，IF～THEN 命令を使って，途中からループを脱出する方法はよく使われるテクニックです．

また，この他にも，GOSUB 命令を使い，一旦ループから脱出して，RETURN 命令で再びもどり，実行を継続することも可能です．

次のプログラムは，FOR～NEXT 間に，何も実行するものがない例です．

```
10  LOCATE 8,2:PRINT "HIT!"
20  FOR I=0 TO 50
30  NEXT I
40  CLS
```

```
50　FOR I=0 TO 50
60　NEXT I
70　GOTO 10
```

20〜30行および50〜60行の FOR〜NEXT 間には，処理すべきステートメントがありませんが，このような場合でも，終了条件を満たすまでFOR〜NEXT命令を実行します．

ちょっと，無意味な命令に見えますが，時間かせぎをしたいような場合によく使い，〝待ちループ〞などと呼んでいます．

このプログラムでは，〝HIT！〞という文字を一定時間表示させ，一定時間消した後，再び一定時間表示させることをくり返しています．

終値を大きくすれば，それだけ，待ち時間が長くなります．

サンプル・プログラム

● 〝*〞印を1つずつ増やして表示させるプログラム

```
10 CLS
20 N=1
30 FOR I=1 TO N
40 PRINT "*";
50 NEXT I
60 PRINT
70 N=N+1
80 IF N=>20 THEN END  ELSE 30
```

このプログラムは，FOR〜NEXT命令の終値を1つずつ増やして，“*”印の表示個数を1つずつ増やしています．右の図が表示例です．

```
*
**
***
****
*****
******
```

参　照　IF〜THEN〜ELSE

プログラム・コマンド

GET　(get：ゲット)

機　能	カセットテープに記録したデータを変数データに読み込む
書　式	GET　変数〔,変数〕 GET　"ファイル名"変数〔,変数〕

GET 命令は，PUT 命令でカセットテープに書き込んだデータを，変数に読み込む命令です．

上記の書式のように，ファイル命令を省略することができますが，省略した場合は，カセットテープ（MT）から送られてくる最初に出現したデータファイルを読み込みます．また，変数をカンマ（，）で区切ることによって，データを順番に異なる変数に読み込むことができます．

ただし，数値変数にデータを読み込む場合には，データ先頭のスペースは無視されます．

使い方

次のプログラムは，変数A，B，C，Dの値を，ファイル名"TEST"をつけて，カセットテープに記録します．

```
10  REM PUT MT
20  A=10:B=20:C=30:D=40
30  PUT "TEST" A, B, C, D
40  END
```

次に，GET 命令を使用して，カセットテープの"TEST"というファイルから，データを読み込む例です．

```
10  REM GET MT
20  GET "TEST" E, F, G, H
30  PRINT E;F;G;H
40  END
```

この例では，前出のA，B，C，Dのデータが，E，F，G，Hの変数に読み込まれます．ここで，大切なことは，カセットテープには，A，B，C，Dという順番で記録されている点です．

したがって，カセットテープを再生した場合には，A，B，C，Dの順にデータが送られてくることになり，AのデータはEに，BのデータはFに……のような順に代入されます．

もし，カセットテープに記録した変数データより多い変数で GET すると，データが足らなくなり，DA エラーとなります．

サンプル・プログラム

● カセットテープに名前とその人のデータを記録し，名前を入れると，その人のデータを読み込み，表示するプログラム．

```
 10 CLEAR
 20 DIM N$(10),D$(10)
 30 INPUT "PUT:O GET:I END:E";M$
 40 IF M$="O" THEN GOSUB 100
 50 IF M$="I" THEN GOSUB 300
 60 IF M$="E" THEN END
 70 GOTO 30
100 I=0
110 INPUT "NAME:";N$(I)
120 PUT N$(I)
130 IF N$(I)="0" THEN 180
140 INPUT "DATA:";D$(I)
150 PUT D$(I)
160 I=I+1
170 IF I<=10 THEN 110
180 RETURN
300 J=0
310 INPUT "NAME:";A$
320 GET N$(J)
330 IF N$(J)="0" THEN PRINT
    "NOT FOUND ":GOTO 380
```

```
340 GET D$(J)
350 IF A$<>N$(J) THEN 370
360 PRINT A$;"=";D$(J):GOTO 380
370 J=J+1:GOTO 320
380 RETURN
```

このプログラムは，2つのサブルーチンで構成されています．

100〜180行が，カセットテープへのデータの記録サブルーチン，300〜390行がカセットテープからデータを読み込み，データを検索するサブルーチンです．

データは，配列変数 N$（I）に名前, D$（I）にその人のデータを入れ，この2つの変数がペアになって，書き込んだり，読み込んだりされます．

30行のメニューで，"O" を入力しますと，100行からの記録サブルーチンが実行され，次々に名前とデータを入力します．最大11個までのデータを入力できますが，最後には，名前に必ず "0"（データの終りを示すため）を入力する必要があるため，実際には，最大10個のデータが記録できます．

30行のメニューで "I" を入力すると，探し出したい人の名前を入力すると，カセットテープから，名前とデータを読み込み，検索します．

その人の名前を探し出したら，名前とデータを表示して，30行のメニューにもどります．

メニューで "E" を入力すれば，プログラムは終了します．

参　照　PUT，SAVE

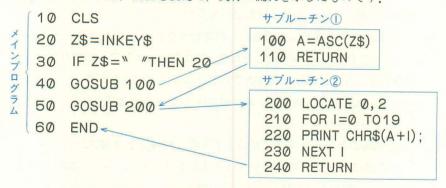
GOSUB/RETURN (gosub/return：ゴーサブ/リターン)

機　能	指定された分岐先へサブルーチン分岐する
書　式	GOSUB　行番号　　1≦行番号<10000 GOSUB PROG n　　0≦n<10 　　　　　　　　　　（ RETURN　　（nはプログラムエリア番号）

GOSUB 命令は，変数や数式で示される行番号のサブルーチンプログラムへの分岐を行ない，RETURN命令は，サブルーチンプログラムからメインプログラムへの復帰を行ないます．

上記の書式のように，PB−700では，同一プログラムエリア内のサブルーチンへの分岐の他に，他のプログラムエリアに書き込まれているサブルーチンプログラムへの分岐および復帰も可能です．

使い方

次のプログラムは，配置を変えて，実行の流れを示したものです．

10行～60行がメインプログラムです．100行～110行が1つのサブルーチンプログラム，200行～240行も1つのプログラムです．

上記の矢印のように，RETURN 命令を実行すると，GOSUB命令の次に復帰し，実行を続けます．

前述のプログラムを多少変形して，次のように書くこともできます．

メインプログラム
```
10  CLS
20  Z$=INKEY$
30  IF Z$="  "THEN 20
40  GOSUB 100
50  END
```

サブルーチン①
```
100  A=ASC(Z$)
110  GOSUB 200
120  RETURN
```

サブルーチン②
```
200  LOCATE 0,2
210  FOR I=0 TO 19
220  PRINT CHR$(A+I);
230  NEXT I
240  RETURN
```

プログラムの内容は，前述のものと全く同じですが，今度は，サブルーチン①の中にサブルーチン②が含まれる二重構造になっています．

GOSUB/RETURN命令では，このような入れ子（ネスティング）は12段まで許されています．

GOSUB/RETURN命令を使うときには，次のような点にも注意してください．

(1)　1つのサブルーチンプログラムには，RETURN命令はいくつあってもかまわないが，必ず1つはなければならない．

(2)　GOSUB命令を実行した場合に，指定行がない場合はULエラーとなる．

(3)　GOSUB PROG n を実行した場合，プログラムエリアnにプログラムが書かれていないときは，現在のプログラムエリア内においてULエラーとなる．

(4)　RETURN命令実行時に，それ以前にGOSUB命令が実行されていないときはGSエラーとなる．

(5)　GOSUBのネスティングは12段まで可能（13段以上はNOエラー）

(6)　GOSUBで指定する行番号やプログラムエリアナンバーに小数部を含む場合は，小数部を切り捨てた上で実行される．

サンプル・プログラム

●スロットゲームプログラム

```
10 CLS
20 GOSUB 200
30 GOSUB 300
```

```
 40 GOSUB 400
 50 LOCATE 3,0
 60 PRINT "TOKUTEN";N
 70 LOCATE 0,0:END
200 X=INT(10*RND)
210 Y=INT(10*RND)
220 Z=INT(10*RND)
230 RETURN
300 LOCATE 6,2
310 PRINT X;"  ";Y;"  ";Z;
320 RETURN
400 IF X=Y THEN IF Y=Z THEN
    N=100:RETURN
410 IF X=Y THEN N=40:RETURN
420 IF Y=Z THEN N=30:RETURN
430 IF X=Z THEN N=20:RETURN
440 N=0
450 RETURN
```

このプログラムは，3ケタの数字が表示され，3ケタとも同じ数字が表示されると，得点が100点，2ケタだけが同じ数字の場合は，その位置によって,40点，30点，20点に分かれています．

3ケタともバラバラの場合は，0点と表示されます．

200〜230行が，RND関数で3つの数字を発生するサブルーチン,300〜320行が，その数字を画面の中央に表示するサブルーチン，400〜450行が，得点をチェックするサブルーチンです．

10〜70行のメインプログラムは，各サブルーチンを制御し，得点を表示するだけです．

参照 GOTO

プログラム・コマンド

GOTO

(go to：ゴートゥー)

機　能	指定する分岐先へ無条件に分岐する
書　式	GOTO 行番号　　1≦行番号<10000 GOTO PROG n　　0≦n<10 n：プログラムエリアナンバー

GOTO命令は，変数や数式で指定された行番号，あるいはプログラムエリア番号のプログラムの先頭行へ，無条件に分岐します．

指定する行番号，あるいは指定するプログラムエリアにプログラムがない場合にはULエラーとなります．

また，行番号，あるいはプログラムエリア番号に小数部を含む場合は，小数部を切り捨てて実行されます．

書式例を示します．

	行　番　号	プログラムエリア
数値定数	GOTO 500	GOTO PROG 4
数値変数	GOTO N	GOTO PROG N
数　式	GOTO N*5	GOTO PROG N/10

サンプル・プログラム

●簡易デジタル時計プログラム

```
10 INPUT "H=";H,"M=";M,"S=";S
20 CLS
30 IF H<10 THEN LOCATE 8,0:PRINT H;":
   ";:GOTO 50
40 LOCATE 7,0:PRINT H;":";
50 IF M<10 THEN LOCATE 12,0:PRINT M;"
   :";:GOTO 70
60 LOCATE 11,0:PRINT M;":";
70 IF S<10 THEN LOCATE 16,0:PRINT S;"
```

```
         :";:GOTO 90
 80 LOCATE 15,0:PRINT S;
 90 FOR I=2 TO 72:NEXT I
100 S=S+1
110 IF S>=60 THEN 200
120 GOTO 70
200 S=0:M=M+1
210 IF M>=60 THEN 300
220 GOTO 50
300 M=0:H=H+1
310 IF H>=24 THEN H=0
320 GOTO 30
```

このプログラムをRUNすると，H（時），M(分)，S(秒）を聞いてきますので，現在の時間を入力し，⏎キーを押すと同時に，秒単位で時間を表示します．あらかじめ，進ませたH，M，Sを入れておき，正確な時計と合わせて　キーを押してください．

このプログラムは，マイコンの内部時計を利用しているわけではないので，正確な時計とはいえませんが，およその時間を知ることは可能です．

進み勝ちのときには，90行のFOR命令の終値を増して調節します．また，遅れる場合は，終値を減らして調節します．

このプログラムには，いたるところにGOTO命令が使われています．

120行，220行，320行のGOTO命令によって，プログラムは無限ループを作っています．

参　照　GOSUB/RETURN，IF～THEN～ELSE

プログラム・コマンド

IF～THEN～ELSE (if～then～else：イフ～ゼン～エルス)

機　能	IF後の条件により,THENもしくはELSE以降を実行する
書　式	IF〈条件式〉THEN {行番号/コマンド} ELSE {行番号/コマンド} 行番号として数式を用いることはできない

GOTO命令の無条件分岐に対して，IF～THEN命令は条件分岐します．

条件式が真の場合，THEN文を実行し，偽の場合は，ELSE文を実行します．

THEN文およびELSE文中には，分岐する行番号あるいはコマンド文を入れることができます．

IF文においては，条件式が成立しない（偽）ときは，（THENの次の行番号へ分岐せず）ELSE の後の行番号へ分岐または，コマンドを実行するということですが，ELSE 文がない場合には，実行は次の命令または行へ移ります．さらに，コロン(：)を用いると，THEN 文，ELSE 文に，次のようなマルチステートメントが可能です．

IF〈条件式〉THEN～：～：～ELSE～：～：～

THEN文　ELSE文

上記の THEN 文あるいは ELSE 文中に，次のようにIF文を入れることができます．

条件式①が真，②が偽のとき実行

IF〈条件式①〉THEN　IF〈条件式②〉THEN～ELSE～ELSE～

条件式①②とも真のとき実行

条件式①が偽のとき実行

使い方

次に，IF～THEN～ELSE命令の使い方の事例を示します．三辺の値を代入して，それが三角形であるかどうかを判断します．

①
```
10 INPUT "A=";A, "B=";B, "C=";C
20 IF A+B>C THEN 40
30 GOTO 50
40 IF ABS(A-B)<C THEN 60
50 PRINT "NOT TRI":GOTO 10
60 PRINT "TRI":GOTO 10
```

②
```
10 INPUT "A=";A, "B=";B, "C=";C
20 IF A+B>C THEN IF ABS(A-B)<C THEN 40
30 PRINT "NOT TRI":GOTO 10
40 PRINT "TRI":GOTO 10
```

③
```
10 INPUT "A=";A, "B=";B, "C=";C
20 IF A+B>C THEN IF ABS(A-B)<C THEN 40
30 PRINT "NOT TRI":GOTO 10
40 IF A=B THEN IF B=C THEN PRINT "E.TRI":GOT
   O 10 ELSE PRINT "I.TRI":GOTO 10
50 IF B=C THEN PRINT "I.TRI":GOTO 10
60 IF A=C THEN PRINT "I.TRI":GOTO 10 ELSE
   PRINT "TRI":GOTO 10
```

①の解説：三角形の条件は，２辺を足した値が他の一辺より大きく，かつ２辺の差が他の一辺よりも小さいものです．この２つの条件が成立するかどうかを，20行と40行で判断しています．ですから，ひとつの条件でも成立しなければ，30行で50行の "NOT TRI"（三角形ではない）へ分岐させます．20行が成立しても，40行が成立しなければ，次の行を実行します．

②の解説：①の２つの条件を，20行でいちどに行なっています．

```
20 IF A+B>C THEN IF ABS(A-B)<C THEN 40
      └─┬─┘         └────┬────┘      └┬┘
       式①              式②       式①が真，式②が真
                                   のとき，40行へ
```

③の解説：このプログラムは，②のプログラムにさらに，正三角形(E.TRI)であるか，二等辺三角形(I.TRI)であるかの判断を加えました．40行で，A＝BかつB＝C，つまり三辺が等しければ，正三角形(E.TRI)を表示し，B＝Cが成立しなければ，二等辺三角形(I.TRI)を表示します．50行は，A≠BでB＝Cの場合の二等辺三角形であることの表示，60行は，A＝Cのときの二等辺三角形であることの表示と，A≠B，B≠C，A≠Cのときの三角形であることの表示です．

```
40 IF A=B THEN IF B=C THEN PRINT "E.TRI" : GOTO
      ─①─          ↑      ─②─            ↑
   式①が真ならば───┘   式②が真ならば────┘
  10  ELSE PRINT "I.TRI": GOTO 10
      └────┘        ↑
       ②が偽ならば─┘
```

サンプル・プログラム

＊画面にドットで模様を描くプログラム

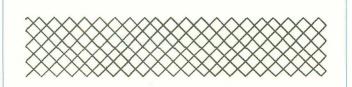
```
 10 CLS
 20 X=0:Y=0:N=1:M=1
 30 DRAW(X,Y)
 40 X=X+N:Y=Y+M
 50 IF X>=159 THEN N=-1
 60 IF Y>=31 THEN M=-1
 70 IF X<=0 THEN N=1
 80 IF Y<=0 THEN M=1
 90 IF X=1 THEN IF Y=31 THEN BEEP
    1:END
100 GOTO 30
```

このプログラムは，画面のドット座標（X，Y）の値が，画面の限界にあるかどうかを，IF〜THEN命令で判断し（40行〜80行），座標のX，Yの値を増やすか，減らすかを，制御しています．

そして，X，Yの値がX＝1，Y＝31となったとき（90行），BEEP音を出し，プログラムを終了します．

下記の図は，実行結果を示したものです．

■実行例

プログラム・コマンド

INPUT (input：インプット)

機　能	キーボードからのデータ(数値,文字)を指定した変数に入力する
書　式	INPUT 変数〔,変数〕 INPUT "メッセージ文", 変数〔,"メッセージ文",変数〕 INPUT "メッセージ文"; 変数〔,"メッセージ文"; 変数〕

INPUT命令は，入力命令の代表格で，キーボードからのデータを変数に入力するための命令です．以下に，INPUT文の基本的な型をあげておきます．

例1　INPUT A

例2　INPUT X, Y, Z

例3　INPUT "NENREI", A

例4　INPUT "NAMAE"; A$

例5　INPUT "X="; X, "Y="; Y

解　説

INPUT命令が実行されると，パソコンは入力要求のメッセージを表示して，データの入力を待ち続けます．

例えば，上の例1が実行されると，次のように"？"が表示され，その右側にカーソルが点燈します．これで，データを入力する準備が整ったわけです．

INPUT命令実行時の表示

```
Ready P0
?_(カーソル)
```

データの入力は，キーボードを押して行ないます．そして，データの最後には必ず ENTER キー，または ⏎ キーを押します．これではじめて，データがパソコンに入力されたことになります．つまり，INPUT文実行中は ENTER キーは，⏎ キーと同じ働きをするということですので，これも覚えておいて下さい．

■INPUT文で使える変数は，次のとおりです

〔例〕

数値変数………INPUT X
文字変数………INPUT X$（7文字まで入力可能）
登録変数………INPUT XY
………INPUT XY$（16文字まで入力可能）
配列変数………INPUT X!(i), X!(i, j) 半精度数値配列
INPUT X(i), X(i, j) 単精度数値配列
INPUT A$(i), INPUT A$(i, j) 文字配列

使い方

(1) 数値を入力するINPUT文

簡単なプログラムで，INPUT文の使い方と働きをみてみましょう．

```
10  INPUT A ……入力命令，変数Aにデータを入力する
20  PRINT A ……出力命令，変数Aの内容を表示
30  GOTO 10……分岐命令，プログラムの実行を10行目に移す
```

プログラムを入力したら，RUN ⏎ または SHIFT PO/∅ と入力して下さい．"?"が表示されたはずです．ここで，3.6⏎と入力してみましょう．正しく入力されていれば，PRINT文によって，下のように同じ数がもう一度表示されるはずです．

CLS
SHIFT PO/∅　3.6 ⏎

```
?3.6
 3.6
?_
```

なお，INPUT文で数値を入力する場合に限って，数値データでなく計算式による入力を行なうことができます．

前のプログラムを使って，これを確認してみましょう．

SHIFT P0 CLS 100—20/5 ↵

```
?100-20/5
 96
?_
```

（2）文字を入力するINPUT文

(1)のプログラムを変えて，文字を入力するようにしてみましょう．プログラムは次のようになります．

```
10  INPUT A$
20  PRINT A$
30  GOTO 10
```

文字を入力する場合には，このように文字変数を使います．実行させると，数値の入力の場合と同じように〝?〞を表示して，入力を要求します．そこで，文字列ＡＢＣを入れてみますと，次のような表示となります．

```
?ABC
ABC
?_
```

仮にここで，数値123を入力した場合，123が表示されますが，これは文字列としての「123」であり，数値ではありませんので注意が必要です．

なお，文字変数にデータを入力する際には，入力する文字列を〝〞（ダブルクォーテーション）で囲む必要はありません．〝〞を使用した場合には，〝〞自身も文字データとして入力されてしまいます．

■INPUT文で各変数に入力できるデータについて

①数値変数…………$\pm 1 \times 10^{-99} \sim \pm 9.999999999 \times 10^{99}$と０
数値の演算式〔例：200×（５＋２）〕
Ａ～Ｚの数値変数（固定変数）
登録変数
配列変数

つまり，変数も含めた数式であることです．

②文字変数

a.固定文字変数……… 7文字までの文字，記号

b.登録文字変数………16文字までの文字，記号

c.配列変数……………指定した文字数（最大79文字）までの文字，記号

（3） 2つ以上の変数を入力するINPUT文……例2

INPUT文には，2つ以上の変数を書くことができます．それは，例2で示したような型で，下のように2つ以上のINPUT文をひとつにまとめたものということができます．

```
10  INPUT X ┐
20  INPUT Y ├ ──→ 10 INPUT X, Y, Z
30  INPUT Z ┘
```

変数をカンマで区切る

このINPUT文を実行すると，最初に"？"が表示され，Xの値の入力を要求してきます．Xの値を入力すると続いてYの値，Zの値と順に入力を要求します．Zの値を入力すると，このINPUT文は終了です．

仮に，X＝123，Y＝456，Z＝789を入力した場合，最終的な表示は次のようになります．

入力	表示
CLS RUN ↵	RUN
1 2 3 ↵	?123
4 5 6 ↵	?456
7 8 9	?789

この型のINPUT文では，

```
10 INPUT A$, X
```

のように，数値変数，文字変数など変数はどのような組み合せ，順序でもかまいません．ただし，変数と変数の区切りは，必ず，（カンマ）を使用します．

（4）メッセージを表示するINPUT文……例3，例4

INPUTと変数の間に，"　"（ダブルクォーテーション）で囲んだ文字列を入れると，その文字列を表示します．これをメッセージ文と呼びますが，これによって入力すべきデータの名前を明示させれば，入力ミスを少なくすることができます．

では，例3のINPUT文を実行させてみましょう．

10 INPUT "NENREI";A …………例3

これを実行させると，

CLS
SHIFT P0

```
NENREI?_
```

のようになります．次にこのINPUT文を下のように変えてみます．

10　INPUT "NENREI", A

これを実行させてみましょう．

CLS
RUN ↵

```
RUN
NENREI_
```

このように，メッセージ文に続けて"？"（入力要求表示）を出したい場合は，メッセージ文と変数の間の区切り符号に；（セミコロン）を使います．

このINPUT文をさらに発展させて，2つ以上の変数を書くことができます．

10　INPUT "SINCHO=" ; X, "TAIJU=" ; Y

これは，次のような2つのINPUT文を，ひとつにまとめたものと考えることができます．

10　INPUT "SINCHO=";X
20　INPUT "TAIJU=";Y
→10 INPUT "SINCHO=";X, "TAIJU=";Y

これを実行すると，次のようになります．

```
SINCHO?175
TAIJU?65_
```

■メッセージ文に使う文字列の文字数

〃 〃で囲んだ文字列の文字数は，行番号，INPUT文を含めて79文字までです．

プログラム・コマンド

LET （let：レット）

機　能	変数にデータを代入する
書　式	LET 変数＝式

解　説

LETは，代入文の頭につけるコマンドです．一般には，次のような形で使用されます．

（例１） LET A＝10　　　LET A$＝"GAME"

（例２） LET X＝SIN (S－PI/4)　　LET X$＝A$＋B$

代入文は，右辺のデータを左辺の変数に代入せよということですが，数値変数には数値式が，文字変数には文字式が対応します．この対応が正しく行なわれていない場合は，TMエラーを表示します．

■数値変数への代入値の許容範囲

数値変数に10^{100}以上の数値を代入することはできない．

■文字変数への文字数の許容範囲

左辺が固定文字変数のとき——7文字まで

左辺が登録文字変数のとき——16文字まで

左辺が文字配列変数のとき——79文字までの指定による．

■LETは省略して使うことができる

10　LET　A＝1　→　10　A＝1　と同じです．

プログラム・コマンド

LOCATE (locate：ロケイト)

機　能	カーソルの位置を指定する
書　式	LOCATE x, y　　0≦x<20, 0≦y<4

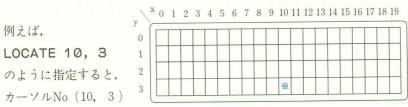
解　説

PB−700の表示画面には，下のように20×4の表示位置があります．通常は，PRINT文を実行すると，画面の左端から表示していきますが，このLOCATEコマンドを使うと，表示位置を自由に変えることができます．

例えば，

LOCATE 10, 3

のように指定すると，

カーソルNo（10，3）

の位置が指定されたことになります．ただし，Y座標に3を指定して表示させますと，改行後スクロールしますので，スクロールさせたくない場合はLOCATE 19,3（右下隅）以外は出力要素の最後に；（セミコロン）をつけてください．

使い方

リストA

```
10  X=1
20  X=X+1
40  PRINT "X=";X
50  GOTO 20
```

リストB

```
10  X=1
20  X=X+1
30  LOCATE 0,0
40  PRINT "X=";X
50  GOTO 20
```

リストAを実行すると，次のような表示になります．

```
X=2
X=3
X=4
X=5
```

⇧ 数が画面の下から次々と現われて，画面の上に消えていく

リストBのように，30行目を追加すると，画面の左上隅でX＝数字の数字の部分だけがどんどん増えていくような表示の形に変わります．

プログラム・コマンド

PRINT/LPRINT (print/l-print：プリント/エルプリント)

機能	PRINT ：画面への出力を行なう LPRINT：プリンタへの出力を行なう
書式	PRINT 式〔, 式〕 PRINT 式〔; 式〕 LPRINT 式〔, 式〕 LPRINT 式〔; 式〕

PRINTとLPRINT命令は，画面に出力するか，プリンタへ出力するかの違いで，他はほぼ同じ働きをします．

ただし，TAB関数を組合せて使うときは，少し違いが生じます．

使い方

PRINT文を使えば，数値はもちろんのこと，文字や数式など，あらゆる形のデータを表示させることができます．

たとえば，次のように使います．

```
PRINT 1.414 ↵
PRINT A*B-2 ↵
```

……A,Bは変数なので，計算結果が出力される

ここに示したようなPRINT文を実行すると，データを表示した後，改行が行なわれます．たとえば，次のプログラムを実行してみます．

```
10 A=0：B=3
20 PRINT 1.414
30 PRINT A*B-2
40 INPUT C
```

すると，画面への表示は，次のようになります．

CLS
RUN ↵

```
RUN
 1.414
-2
?_
```

PRINT文は，カンマ（，）を用いることによって，複数の式や文字列を表示させることができます．

```
10  A=0:B=3
20  PRINT 1.414, A*B-2
30  INPUT C
```

これを実行すると，前記の表示と同様に，１つのデータずつ改行されて出力されます．

```
RUN
 1.414
- 2
?_
```

PB−700の画面は４行ですが，４行目に出力すると，各行がロールアップされます．

```
10  PRINT 1, 2, 3, 4
20  END
```

```
4                RUN  ⎫
                  1   ⎬ はロールアップされる
Ready P0          2   ⎪
                  3   ⎭
```

前述のカンマ(，)で区切ると，改行されて出力しましたが，セミコロン(；)により，複数の式や文字列を区切ると，次のように，続けて表示されます．

```
10  A=0:B=3
20  PRINT 1.414 ; A*B-2
30  END
```

```
1.414-2
Ready P0

_
```

この方法を利用すると，次のように，メッセージ付きのわかりやすい表示ができます．

```
10  A=0 : B=3
20  PRINT "KOTAE=" ; A*B-2
```

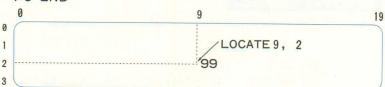
```
30 END
```

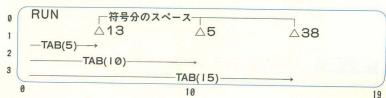
```
KOTAE=-2
Ready P0

_
```

これまでの出力例でもおわかりのように，文字出力，数値出力とも左づめで出力されます．ただし，数値出力は符号分1桁も合わせて出力するため，正の数値の場合は，＋符号を省略した1桁がスペースとなります．

ところで，PRINT文には，表示位置や表示書式を指定できる，LOCATE 命令やTAB，USINGなどの関数が用意されています．

```
10 CLS
20 A=1
30 LOCATE 9, 2
40 PRINT A
50 A=A+1
60 IF A<100 THEN 30
70 END
```

上記の表示のように，キャラクター画面の（9,2）の座標の位置に1～99の数値が連続的に表示されました．（9,2）の座標点には，符号分スペースが表示されています．

```
10 CLS
20 A=13:B=5:C=38
30 PRINT TAB(5);A;TAB(10);B;TAB(15);C
40 END
```

このように，PRINT文とTAB関数を合わせて使うと，TAB関数で指定した位置から，表示されます．

また次のようにUSING関数のフォーマットに従って，書式をそろえて，表示することができます．

```
10  A=3.1415：B=31.415：C=314.15
20  PRINT USING "###.##"；A
30  PRINT USING "###.##"；B
40  PRINT USING "###.##"；C
50  FOR I=1 TO 1000：NEXT I
60  END
```

```
  3.14
 31.42………小数点第3位で四捨五入されている
314.15
```

サンプル・プログラム

●書式をそろえて，円の半径，円の面積，円周を表示するプログラム

```
10 CLS
20 PRINT " R";TAB(7);"S";TAB(16);"L"
30 FOR R=1 TO 9
40 S=PI*R^2
50 L=2*PI*R
60 PRINT R;
70 PRINT "  ";:PRINT USING"###.##";S;
   :PRINT "   ";:PRINT USING"##.##";L
80 NEXT R
90 END
```

参　照　LOCATE, TAB, USING

プログラム・コマンド

PUT (Put：プット)

機　能	変数データをカセットテープに書き込む
書　式	PUT　変数〔，変数〕 PUT　"ファイル名"変数〔，変数〕

解　説

PUT命令は，変数データをカセットテープに記録する命令です．記録するデータファイルの形式は，アスキー形式です．このデータファイルは，GET命令で読み込みます．

(1)　PUT A〔，B，C……〕

(2)　PUT "URIAGE" A〔，B，C……〕

上記の書式のように，ファイル名を省略することができますが，省略した場合は，これに対応するGET命令には，ファイル名はつけません．また，変数をカンマ(，)で区切ることによって，一つのPUT命令で，複数の変数を，データファイルとして記録することができます．この場合データが記録される順番は，PUTの次に書いた変数のうち，前のが先になります．また，数値データは，USING関数を使用しない場合の画面への表示と，同じ内容が出力されます．

使い方

次のプログラムは配列変数A(0)～A(10)の内容をPUT命令で記録するものです．配列変数にはデータが入っているものとします．

```
10 REM PUT
20 DIM A(10)
30 FOR K=0 TO 10
40 A(K)=K
50 PUT A(K)
60 NEXT K
70 END
```

```
10 REM GET
20 DIM B(10)
30 FOR K=0 TO 10
40 GET B(K)
50 PRINT "B(";K;")=";B(K)
60 NEXT K
70 END
```

次にGET命令を使用して，カセットテープに記録した上のデータをPB-700に読み込む例です．この例では，FOR～NEXTループを使って，Aの配列の内容をBへ移しています．なお，複数のデータを記録するときは，順番に十分気をつけて下さい．

参　照　GET, SAVE

プログラム・コマンド

READ/DATA/RESTORE (read/data/restore：リード/データ/リストア)

機　能	READ：DATA文で格納されたデータを読み込む DATA：READ文で参照するデータ(定数, 文字)を格納する RESTORE：DATA文の実行順序を変更する
書　式	READ　変数〔,変数〕 DATA　データ〔,データ〕〔,"文字データ"〕 RESTORE RESTORE　行番号（数式）　1≦行番号<10000

READ文は，DATA文と組合せて使います．

READ文が実行されると，DATA文からデータを変数に1対1で読み込んでいきます．

使い方

次の例は，最も簡単なREAD/DATA文のプログラムです．

```
10  READ A
20  READ B$
30  PRINT A;B$
40  DATA 7,"B"
50  END
```

このプログラムを実行すると，変数Aには7が，文字変数B$には"B"が代入されます．変数の型とデータの型は一致していなければなりません．

READ文は，変数をいくつも続けて書くことができますので，10行，20行は，次のように，1行に置き換えることができます．

```
10  READ A, B$
```

また，文字データは，上記のように，ダブルクオーテーション（"）で囲っても，次のように囲わなくてもかまいません．

```
40  DATA 7, B
```

ただし，この場合は，先頭のスペースが無視されますので，スペースが必要な場合は，ダブルクオーテーションで囲わなければなりません．

（例）　DATA 7，"␣","␣A"，␣C

└──このスペースは無視される．

また，文字データの中には，上記の形を除いて，ダブルクオーテーション（"）やカンマ（，）は書くことができます．

READ文の変数とDATA文は１対１に対応することが必要ですが，それぞれの文の中に，変数やデータをいくつ並べるかは自由です．

```
10 CLEAR
20 DIM C(10)
30 READ A, B
40 FOR I=1 TO 10
50 READ C(I)
60 NEXT I
70 DATA 1,2,3
80 DATA 4,5,6,7,8,9
90 DATA 10,11,12
```

このプログラムを実行すると，次のように，各変数にデータが代入されます．

```
A  B  C(1)  C(2)  C(3)………C(8)  C(9)  C(10)
↓  ↓   ↓     ↓     ↓         ↓     ↓     ↓
1  2   3     4     5 …………10    11    12
```

もし，READ文で読み込む変数の数よりも，データの数が少ない場合には，DAエラーとなります．しかし，データの方が多い場合はエラーが発生することなく，残りのデータは無視されます．

また，DATA文は，READ文より前にあってもかまいません．

次に，RESTORE文を使うと，READ文で読むDATA文を指定することができます．

RESTORE文には，行番号を書く形式と，書かない形式があります．

行番号を書かない場合は，RESTOREを実行すると，次のREAD文は最初のDATA文からデータを読みます．

```
10  READ A, B
20  RESTORE
30  READ C, D
40  PRINT A ; B ; C ; D
50  DATA 7, 2
60  END
```

このプログラムを実行すると，A＝7，B＝2，C＝7，D＝2 のように代入されます．

行番号を指定すると，使うデータ文を指定することができます．

```
10  RESTORE 50
20  READ A, B
30  PRINT  A ; B
40  DATA 3.7, 6.5
50  DATA 7.1, 9.3 ………10行のRESTOREで指定されたDATA文
60  DATA 5, 10.2
70  END
```

このプログラムを実行すると，A=7.1　B=9.3が代入されます．

なお，RESTORE 文の指定行番号には，変数や数式を使うことも可能ですが，変数や数式の値がDATA 文のある行番号となるようにしてください．

次に，複数のプログラムエリアを動きまわるプログラムで，READ文を使う場合には，次のような注意が必要です．

```
P0  10  READ A,B             P1  10  READ X, Y
    20  GOTO  PROG 1             20  PRINT X ; Y
    30  DATA 1,2,3,4             30  DATA 71,65
                                 40  END
```

このプログラムを実行すると，表示されるデータは，71，65ではなく，3，4となります．つまり，GOTO PROG 1で，プログラムの実行は，P１へ移っ

てはいるのですが，DATA文だけは，まだP0のものが使われているのです．

このことは，サブルーチンプログラムで，メインプログラムのDATA文を使うという場合にも役立ちます．

もし，前述のプログラムで，X，Yに71，65を読み込ませたい場合は，P1のプログラムの先頭に，次のように，行番号を指定します．

```
5 RESTORE 30
```

つまり，RESTORE文には，DATA文を使う順番と，プログラムエリアを指定する2つの働きがあります．

サンプル・プログラム

＊エイリアンの移動プログラム

```
 10 CLS
 20 N=0
 90 FOR I=1 TO 58
100 READ X,Y
110 IF X=0 THEN RESTORE :N=N+17 :GOTO 100
120 IF N=>140 THEN END
130 DRAW(X+N,Y)
140 NEXT I
143 CLS
145 GOTO 90
150 DATA 5,15,6,15,7,15,8,15,9,15,10,1
    5
160 DATA 12,15,13,15,14,15,15,15,16,15
    ,17,15
170 DATA 5,16,10,16,12,16,17,16
180 DATA 8,17,9,17,10,17,11,17,12,17,1
    3,17,14,17
190 DATA 8,18,9,18,11,18,13,18,14,18
200 DATA 9,19,10,19,11,19,12,19,13,19
```

```
210 DATA 8,20,9,20,10,20,12,20,13,20,1
    4,20
220 DATA 9,21,10,21,11,21,12,21,13,21
230 DATA 9,22,10,22,12,22,13,22
240 DATA 6,23,7,23,8,23,9,23,10,23,12,23
    ,13,23,14,23,15,23,16,23
250 DATA 0,0
```

このプログラムは，上記の表示例のようなキャラクターが，画面の左から右へ移動します．

150行～250行までのDATA文は，1つのキャラクターを描くためのドットの座標データです．

1つのキャラクターを描くと143行で消し，110行でデータをRESTOREして，位置を移動して同じキャラクターを描きます．

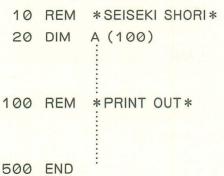
これを繰り返し，画面の右端までくると，プログラムは終了します．

プログラム・コマンド

REM　(remark：リマーク)

機　能	プログラムに注釈をつける
書　式	REM〈注釈文〉

解　説

REMコマンドは，他のコマンドと異なり何も実行しません．REMの後には，自由になんでも書くことができるので，下のように，プログラム中の要所要所にプログラムの説明を書き込み，リストを見たときプログラムの各部分の内容がわかるようにすることができます．

REMの後に書いた文字や記号は，全て注釈とみなされますので，マルチステートメントで，他の命令を続けることはできません．

プログラム・コマンド

STOP　(stop：ストップ)

機　能	プログラムの実行を中断する
書　式	STOP

解　説

プログラム実行中にSTOPコマンドに行きあたると，STOPメッセージを表示して，プログラムの実行が中断します．また，STOPコマンドで停止したプログラムは，CONT命令を入力することにより，停止した次の命令から実行を再開することができます．

使い方

次のプログラムで，具体的なSTOPコマンドの働きをみてみましょう．

```
10 A=1:B=5
20 C=A+B
30 STOP
40 PRINT C
50 END
```

このプログラムを実行すると

STOP P0－30

を表示します．つまり，プログラムエリアP０の30行目で，STOPコマンドによって実行を中断していることを示しています．

この状態のとき，次のようなことができます．

A ENTER ──→ 1　Aの値を表示する

C ENTER ──→ 6　Cの値を表示する

つまり，変数の内容を見ることができるわけです．また，

C＝0 ↵

と入力して，変数に任意の値を代入することもできます．

実際には，プログラムのデバッグ中に，動作の怪しい部分にSTOP命令を挿入

してプログラムを強制的に止め，変数の内容を確かめるというような使い方をします．

1 STOP命令実行中，次のことを行なっても，CONTにより実行を再開することが可能です．

①マニュアル計算

②変数への代入（LETをつけない代入）

③変数内容の確認

④次のコマンドの実行

ANGLE, BEEP, CLEAR, CLS, DIM, ERASE, PRINT, LPRINT, TRON, TROFF

2 STOP命令実行中，次のことを実行すると，CONTによる実行再開はできなくなります．

①マニュアルコマンドの実行（EDIT, SAVE, LOAD, LISTなど）

②PUT／GETの実行

③エラーを発生させた場合

参照 CONT

プログラム・コマンド

TRON/TROFF

(trace-on ：トレース・オン
trace-off：トレース・オフ)

機　能	プログラムの実行状態を追跡する
書　式	TRON TROFF

解　説

TRON，TROFFコマンドは，プログラムのデバッグ時に使われます．

TRON …トレースモードの設定

TROFF…トレースモードの解除

トレースモードが設定されると，現在実行しているプログラムエリアの番号と，そのプログラムの行番号を下のように表示しながら，プログラムを実行します．

表示　〔0：110〕

└実行中のプログラム行番号

└実行中のプログラムエリアNo.

なお，TRON，TROFFはプログラムコマンドですので，プログラム中に書き込んで使えますが，通常はダイレクトモードで入力し，使用します．

使い方

まず，下のプログラムを入力して，実行させてみて下さい．

```
P0
  10  BEEP 1
  20  GOTO PROG 1

P1
  10  BEEP 0
  20  GOTO PROG 0
```

これをRUNさせてみると，２つのBEEP音が交互に鳴り始めます．ここで，

[BRK]キーを押し，TRON ↵

を入力し，トレースモードにして下さい．入力が終ったら，続いてP0をRUNさせて下さい．下のような表示で次つぎと，現在実行中のプログラムエリアと行番号を表示します．

```
〔0：10〕　　　〔0：20〕
〔1：10〕　　　〔1：20〕
〔0：10〕　　　〔0：20〕
```

なお，BEEP音の間隔がかなり開いていることからもわかりますが，トレースモードでは，実行速度が遅くなります．

また，INPUT文の実行時には〔エリアNo：行番号〕を実行後"?"を表示して停止します．またPRINT文では，結果を表示します．このように，プログラムの実行過程を追跡することができますので，デバッグ時には大変便利なコマンドです．

文字関数

ASC (ascii：アスキー)

機　能	先頭1文字の10進コードを与える
書　式	ASC (“文字列”) または ASC (文字変数)

パソコンで表示される文字や数字，記号(これらをキャラクターという)は，すべてアスキー・コードと呼ばれる背番号をもっています．

たとえば，……

“A”………… 65 (番)　(307ページ・キャラクター・コード表を参照してください)
“B”………… 66 (番)
“6”………… 54 (番)

……のようになっています．

このようなキャラクターの背番号 (コード) を知りたいとき，キャラクター・コード表を見てもよいのですが，“ASC” 関数を使って，直接，パソコンに聞くことができます．

使い方

```
PRINT　ASC (“E”) ↵
```

……とキー入力すれば，キャラクター “E” のアスキー・コード……

```
69
```

……が表示されます．

欲ばって，2つ以上のキャラクター・コードを知りたいと思い……

```
PRINT　ASC (“EF”) ↵
```

……とキー入力しても，結果は，最初に入力した “E” のキャラクター・コードしか表示してくれません．

したがって，長い文字列 (たとえば，“ABCDEF……”) を，頭から次つぎと知りたいときは，MID$ 関数を使ったプログラムで表示させるとよいでしょう (MID$の項参照)．

サンプル・プログラム

●キャラクター・コードを次つぎと表示するプログラム

```
10 REM ASCII CODE
20 CLS
30 INPUT "WHICH CHARACTER";A$
40 PRINT ASC(A$)
50 GOTO 30
```

このプログラムは，30行でキャラクターを入力すると，40行でそのコードを表示します．50行で再び30行にもどりますので，次つぎとキャラクターを入力して，そのコードを知ることができます．

実行結果は，次のようになります．

実行例

```
WHICH CHARACTER? A
 65
WHICH CHARACTER? T
 84
WHICH CHARACTER? R
 82
WHICH CHARACTER? 7
 55
WHICH CHARACTER? 1
 49
```

このプログラムは，無限にキャラクター入力を要求してきますので，止めるときは，BRK キーを押してください．

参照　　CHR $

文字関数

CHR$ (character$：キャラクター・ダラー)

機　能	アスキー・コードの1文字を与える
書　式	CHR$ (数式)　　0≦数式<256

CHR$は，アスキー・コードを指定することにより，キャラクター（文字，数字や記号）を引き出す関数です．

使い方

```
PRINT  CHR$(66) ↵
```

……とキー入力すると，アスキー・コード66番のキャラクター“B”が表示されます．

2つのキャラクターを一度に知りたいときは……

```
PRINT  CHR$(71);CHR$(80) ↵
```

……のようにキー入力すれば，アスキー・コード71および80に対応するキャラクター“G”および“P”が連続して表示されます．

PB－700でキーからダイレクトに入力できるキャラクターは，数字，英大文字，英小文字，および一部の記号などですが，その他のキャラクター（たとえば，カタカナ，記号，グラフィック記号など）は，このCHR$を用いて表示させます（巻末．キャラクター・コード表参照）．

なお，CHR$で指定できる数字（数式）は 0≦数式< 256 の範囲内で，小数点は無視されます．

サンプル・プログラム

●キャラクター・コード33～254のキャラクターを表示するプログラム

```
 5 V=0
10 FOR I=33 TO 254
20 PRINT I
30 V=V+1
40 FOR J=1 TO 14
```

```
 50 PRINT CHR$(I);
 60 NEXT J
 70 PRINT
 80 IF V<3 THEN 110
 90 K$=INKEY$:IF K$="" THEN 90
100 V=0
110 NEXT I
120 END
```

このプログラムを RUN すると，キャラクター・コード33～35に対応するキャラクターが，14個ずつ表示され，一時中断します．続いていずれかのキーを入力すると，次の3コード分のキャラクターが14個ずつ表示されます．この操作をくり返し，254番のキャラクターの表示が終わると，プログラムは終わります．実行例の一部を下に示します．

```
 33
!!!!!!!!!!!!!!
 34
""""""""""""""
 35
##############
 36
$$$$$$$$$$$$$$
 37
%%%%%%%%%%%%%%
```

参照　ASC

文字関数

VAL (value：バリュー)

機　能	文字列を数値に変換する
書　式	VAL（"文字列"）または VAL（文字変数）

VALは，文字列を数値に変換する関数です．

いささか変った関数ですが，この関数を理解するには，まず，文字と数値の違いについて，説明しておかなければなりません．

次の2つのプログラム例を比べてみてください．

プログラム①

```
10 READ A,B
20 C=A+B
30 PRINT C
40 END
50 DATA3,5
```

プログラム②

```
10 READ A$,B$
20 C$=A$+B$
30 PRINT C$
40 END
50 DATA3,5
```

プログラム①では，数値変数A，Bに，数値データを読み込み，演算の結果をCに代入して，表示させています．

RUNさせた結果は，もちろん……

```
␣8
```

……となります．上記の␣はスペースを表わすために記しておきましたが，ここには"＋"という符号が入るはずのものが省略されています．

一方，プログラム②では文字変数A$，B$に，3，5をそれぞれ読み込みますが，ここでは，数値データとしてではなく文字データとして読み込まれます．文字列演算は足し算のみが可能で，その結果がC$に代入されます．

このプログラムをRUNさせると……

```
35
```

……と表示されます．この結果は，単なる文字列の表示にすぎません．その証拠に，"－"の符号や，"＋"の符号が入る予定の1文字分の空白がありません．

この空白は，重要な意味をもっています．次のプログラム③，④の実行例を比べると，その差がはっきりするでしょう．

プログラム③

```
10 FOR I=1 TO 10
20 READ A
30 PRINT A;
40 NEXT I
50 END
60 DATA3,8,-6,7,21
70 DATA223,18,8,1,0
```

実行例

```
 3 8-6 7 21 223 18 8
 1 0
```

プログラム④

```
10 FOR I=1 TO 10
20 READ A$
30 PRINT A$;
40 NEXT I
50 END
60 DATA3,8,-6,7,21
70 DATA223,18,8,1,0
```

実行例

```
38-672122318810
```

使い方

前述のプログラム②のように，文字変数 A$，B$ に読み込まれた数字を使ってプログラム①のような演算を行なうときは，VAL 関数を使って，次のように変更すれば可能です．

```
10 READ A$,B$
20 C=VAL(A$)+VAL(B$)
30 PRINT C
40 END
50 DATA3,5
```

RUN すると，プログラム①と同様“␣8”が表示されます．

なお，VAL 関数を使用するときは，次の点に注意してください．

① 文字列の中に，数値，小数点，符号（+，－），指数部のE以外の文字が出現した場合には，それ以後の文字はすべて無視されます．
（Eが2度以上出現した場合も，同様に無視されます）

② 文字列の中の先頭のスペースは読みとばす．

③ 文字列の先頭が数値，小数点，符号でない場合，また文字列が符号，小数点のみの場合は0を与える．

④ Eの後に数字が3つ以上ある場合はSNエラーとなる．

サンプル・プログラム

● メニューナンバー1～5の入力要求のとき，どんなキーが押されてもエラーの出ない入力サブルーチン

```
100 Z$=INKEY$:IF Z$="" THEN 100
110 IF Z$<"1" THEN 100
120 IF Z$>"5" THEN 100
130 GOSUB VAL(Z$)*1000
140          ∫
```

このプログラムは，1～5のキーが押されると，1000～5000行の各サブルーチンへ飛び，それ以外のキーが押されると再入力を要求し，エラーは起きません．メニュー選択のときの入力ルーチンとして利用すると便利です．

参照 STR$

文字関数

STR $ (string $：ストリング・ダラー)

機　能	数値を文字列に変換する.
書　式	STR$（数式）

STR$関数は，数値を文字列に変換します.

使い方

プログラム①および②の実行結果はどのようになるかわかりますか.

プログラム①

```
10 A=25:B=30
20 C$=STR$(A+B)
30 PRINT C$
40 END
```

プログラム②

```
10 A=25:B=30
20 C$=STR$(A)+STR$(B)
30 PRINT C$
40 END
```

2つのプログラムは，同じように見えますが，①は“55”と表示され，②は“ 25 30”と表示されます.

プログラム ① は，数式A＋Bの結果をSTR$関数で文字列に変換しているのに対し，プログラム ② は，数値変数A，Bの内容をそれぞれ文字列に変換したのちに，たし算をしています．この違いが，実行結果の違いとして表われています.

サンプル・プログラム

●たし算練習プログラム

```
10 REM TASHIZAN
20 FOR I=1 TO 5
30 X=INT(RND*100)
40 Y=INT(RND*100)
50 Z=X+Y
60 PRINT STR$(X);"+";RIGHT$(STR$(Y),L
EN(STR$(Y))-1);
70 INPUT "=",A
```

```
 80 IF A=Z THEN PRINT "OK" ELSE 60
 90 NEXT I
100 END
```

(RIGHT$, LENの項参照)

このプログラムは，2桁までの整数のたし算プログラムです．

全部で5問出題されますが，RND関数を使っているので，たす数，たされる数に何が出てくるかわかりません．

60行にSTR$関数を使っていますが，ここで問題を表示します．

別に，STR$を使わずに……

```
60 PRINT  X;"+";Y;
```

……としてもよさそうですが，この方式ですと，次のように，数値の前に，符号分の空白が生じ，不自然になります．

␣15＋␣30＝?

この点が，STR$関数を使うか，使わないかの違いです．

参照　VAL (STR$関数の逆の関数がVAL関数です．)

文字関数

LEFT$ (left $：レフト・ダラー)

機　能	文字列の左から指定文字数取り出す
書　式	LEFT$（文字式，数式） LEFT$（"文字列"，数式） LEFT$（文字変数，数式）

LEFT$は，文字式で表わされる文字列の左から指定した数だけ文字列を取り出す関数です．

使い方

```
10  AB$="LEFT RIGHT"
20  B$=LEFT$(AB$,4)
30  PRINT B$
```

……代入する文字列が7文字以上のため，登録変数(16 文字まで代入できる）を使っていることに注意してください．

RUN すると，"LEFT" が表示されます．つまり，AB$の文字列の左から4文字が取り出されています．

なお，取り出す文字数に0を指定するとヌルが与えられ，存在する文字数以上を指定するとSTエラーとなります．

また，変数や数式でも取り出す文字数を指定することができます．

サンプル・プログラム

●文字列の表示を，順番に増やすプログラム

```
10 AB$="READ LEFT"
20 N=LEN(AB$)
30 FOR I=1 TO N
40 BC$=LEFT$(AB$,I)
50 PRINT BC$
60 NEXT I
70 END
```

実行例

```
R
RE
REA
READ
READ
READ L
READ LE
READ LEF
READ LEFT
```

参照　RIGHT$

文字関数

RIGHT $ (right $：ライト・ダラー)

機　能	文字列の右から指定文字数取り出す
書　式	RIGHT$ (“文字列”)　　　　RIGHT$ (文字変数)

RIGHT$は，文字変数に代入されている文字列を，右から指定した数だけ取り出す関数です．

使い方

```
10  AB$＝“LEFT RIGHT”
20  B$＝RIGHT$(AB$,5)
30  PRINT  B$
```

……文字列が7文字以上であるため登録変数を使っていることに注意してください．

プログラムをRUNすると，“RIGHT”が表示されます．つまり，AB $の文字列を右から5文字取り出したことを示しています．

なお，取り出す文字数に0を指定するとヌルが与えられ，存在する文字数以上を指定するとSTエラーとなります．

また，文字数として変数や数式で指定することもできます．

サンプル・プログラム

●文字列の間に文字列をはさみ込むプログラム

```
10 AB$="ASA YORU"
20 BC$="HIRU"
30 CD$=LEFT$(AB$,3)
40 CD$=CD$+" "+BC$
50 CD$=CD$+RIGHT$(AB$,5)
60 PRINT CD$
70 END
```

このプログラムは，LEFT$とRIGHT$を使い，AB$の文字列の間に，BC$の文字列をはさみ込んでいます．

参照　LEFT$

実行結果

```
ASA HIRU YORU
```

文字関数

MID$ (mid $：ミッド・ダラー)

機　能	指定した位置から右へ指定数だけ文字列を取り出す
書　式	MID$ ("文字列", 数式 1, 数式 2) MID$ (文字変数, 数式 1, 数式 2)

MID$ は，文字変数の文字列を，左から何文字目かを起点として，そこから何文字かを取り出す関数です．ちょうど，LEFT$と RIGHT$ を合わせたような機能をもつ関数です．

MID$ (A$, 3, 2)

A$の文字列 の 左から 3 文字目 ～ 2 文字分 取り出す

使い方

```
10  CLEAR
20  DIM A$(0)*20  ……  文字配列により，20文字まで
                      代入可能（162ページ参照）
30  A$(0)="LEFT␣CENTER␣RIGHT"
40  B$=MID$(A$(0),6,6)
50  PRINT B$
60  END
```

このプログラムを RUN すると，A$(0)の文字列の 6 番目から 6 文字分が取り出され，"CENTER" が表示されます．

MID$ 関数を使うときは，次の点に注意してください．

MID$ (文字式, n, m) とすると……

① n，mの小数部は切り捨てた値となる．
② mが 0 の場合，および抜き出す文字がない場合は，ヌルを与える．
③ ，mが省略された場合は，n 番目以後をすべて与える．
④ mが残りの文字数を超えた場合は，n 番目以後をすべて与える．
⑤ nが文字長より大きい場合は，ヌルを与える．
⑥ nおよびmは，変数や数式を用いることができる．
⑦ nが 1 ≦ n ＜ 256，mが 0 ≦ m ＜ 256の範囲を越えた場合はエラー (BS error) となる．

サンプル・プログラム

●文章中にアルファベットの小文字rがいくつあるかを数えるプログラム

```
10 DIM A$(0)*50
20 N=0
30 INPUT "DATA=";A$(0)
40 M=LEN(A$(0))          (LENの項参照)
50 FOR I=1 TO M
60 IF MID$(A$(0),I,1)="r" THEN N=N+1
70 NEXT I
80 PRINT N
90 END
```

このプログラムは，英文を入力して，小文字の“r”がいくつあるかを数え，その数を表示します．

入力できる英文の文字数は，50文字までで，スペースも文字数に含みます．

たとえば，……

Learning␣to␣master␣your␣CASIO␣Personal␣Computer

……のような文章を入力しますと，最初の文字から1字ずつ，“r”と一致しているかどうかをチェックして，一致している数をカウントします．

上記の文章を入力すると，“5”と表示されるはずです．ためしてみてください．

参照　LEFT$, RIGHT$

文字関数

LEN (length：レングス)

機　能	文字列の長さを与える
書　式	LEN（文字式） LEN（"文字列"）または LEN（文字変数）

LENは，文字変数に代入されている文字列の長さ（個数）を与える関数です．

使い方

```
CLEAR ↵
PRINT LEN(A$) ↵
```

……のようにキー入力しますと，“0”と表示されます．

つまり，CLEARコマンドで変数A＄はカラッポになっているから，当然ですね．

```
AB$="CASIO␣COMPUTER" ↵
PRINT LEN(AB$) ↵
```

今度は，“14”と表示されます．

LEN関数で与えられる値は，0～79の範囲内です．

サンプル・プログラム

●文字列を右づめで表示する．

```
10 INPUT AB$
20 L=20-LEN(AB$)
30 LOCATE L,3
40 PRINT AB$;:LOCATE 0,0
50 END
```

このプログラムは文字列を入力すると，画面の右づめで表示されます．入力できる文字数は16文字までです．

文字関数

INKEY$ (in-key$：インキー・ダラー)

機　能	キーボードからの1文字を入力
書　式	INKEY$

INKEY$関数は，INPUT 命令と似た役割をする入力命令の一種ですが，その働きは多少異なっています．

INKEY$関数で，データを入力できるのは，INKEY$が実行しているときだけで，押したキー（データ）を1文字だけ，文字として入力します．キーが押されていなければ，何も入力されない（ヌル）状態で次の命令の実行に移ります．また，データ入力の際，↵キーを押す必要はありません．

INKEY$とINPUTの違いを次に示します．

使い方	INKEY $	INPUT
命令実行時の表示	何も表示しない	?_(入力要求メッセージ，ただし?は表示させないことも可能)
データの入力	命令実行時に押されたキー．(押されていなければ未入力)	↵が押されるまでに入力したデータ
データの種類	文字として1文字	数値，文字など変数の範囲内の桁および文字数
次の命令の実行	即実行（↵不要）	↵が押されるまで中断

使い方

INKEY$は押したキーを文字として扱いますので，一般に……

文字変数＝INKEY$

……のような代入式の形で使います．

```
100  A$=INKEY$
110  IF A$=""THEN 100
120  IF A$="E"THEN END
130  ............................
```

このプログラムは，"E"のキーが押されたら終了し，その他のキーが押されたときのみ，次の命令の実行に移ります．キーが押されなければ，100～110

行をぐるぐる無限に回転します．

なおINKEY$は BRK キーを除くすべてのキーを読み込みますが，SHIFT キーと CAPS キーはヌルとして処理されます．

また，SHIFT キーもしくは，CAPS キーと他のキーが同時に押された場合は，シフトモードもしくは，キャピタルモードにおけるキャラクターが与えられます．ただし，ワンキーコマンドはヌルを与えます．

サンプル・プログラム

●決められた文字数だけを，文字変数に読み込むサブルーチン

```
10 AB$=""
20 FOR I=1 TO 16
30 K$=INKEY$
40 IF K$="" THEN 30
50 IF K$="*" THEN 80
60 AB$=AB$+K$
70 NEXT I
80       ∫
```

文字変数には，決められた範囲の文字数しか代入できません．

INPUT 文で，適当に文字列を入力していくと，決められた範囲の文字数を越えると，UV エラーとなってしまいますが，このプログラムでは，16文字まで入力すると，それ以上は入らず，次の命令（80行以降）へ移ります．

また，16文字以内で止めたいときは，“＊”を入力すると，次の命令（80行以降）へ移ります．上記のプログラムの80行に……

```
80 PRINT AB$
```

……として，ためしてみてください．

```
P0      10  A$=INKEY$
        20  IF  A$="0"  THEN  100
        30  IF  A$="1"  THEN  200
        40  IF  A$="2"  THEN  300
        50  GOTO  10
                  ⋮
```

以上のプログラムの場合 SHIFT P0 で実行させると，0を押し続けている間に10行目の INKEY$ で0 を読み込みますので御注意下さい．

参照　INPUT

表示用関数

TAB (tabulator：タブ)

機　　能	指定された位置までカーソルを移動させる
書　　式	TAB (数式)　　　　0 ≦数式＜80

PRINT 文および LPRINT 文中で使い，カーソルを移動させ，所定の位置へ表示させる関数です．

使い方

```
10  FOR  X=1  TO  5
20  PRINT X; "^2:";
30  PRINT TAB(10);X^2
40  NEXT X
```

このプログラムを RUN させると……

```
△1^2:␣␣␣␣␣△1
△2^2:␣␣␣␣␣△4        △は＋符号の省略されたスペース
        ⋮
    TAB(10)
```

……のように表示されます．

上記のように，TAB(10) と指定すると，10桁目までカーソルを移動し，その直後から次の表示が始まります．

なお，TAB で指定できる値は，0 以上～80未満の値で，この範囲内の値をとるものであれば，変数や数式でもかまいません．

また，小数値は小数点以下が切り捨てられます．さらに，現在の表示位置(カーソルの位置) から指定された位置までの間は，スペースを置きます．

サンプル・プログラム

●アスキーコードと対応するキャラクターを所定の位置に表示する．

```
10 FOR I=33 TO 254
20 PRINT "ASC";
30 PRINT TAB(5);I;
```

```
40 PRINT TAB(13);"CHR";
50 PRINT TAB(19);CHR$(I)
60 NEXT I
70 END
```

このプログラムをRUNさせると，次のように表示されます．

↵キーを押すと，次のコードとキャラクターが同じ位置に表示されます．

表示位置の与え方

1. 行の左端を0として，文字単位でひとつずつ，右に向って増加するように数えます．
2. 現在の位置より左の方が指定されると，次の行へ改行し，その行の先頭から数えた位置が指定されます．
3. 1行の範囲を超えた位置が指定されたとき，現在の行の先頭から数えた位置が指定されます．

参照 USING, PRINT, LPRINT, LOCATE

表示用関数

USING (using：ユージング)

機　能	表示書式（フォーマット）を指定する
書　式	USING "フォーマット文字列"；

USING 関数は，PRINT 文もしくは LPRINT 文中で，数値や文字列などを指定した一定の書式（フォーマット）に統一して表示します．

使い方

数値を何行かにわたって表示させるとき，桁や小数点などがずれて表示されることがあります．USING 文を使うと，次のようにそろえることができます．

```
10  A=18.5
20  B=2.67
30  C=135.78
40  PRINT USING "#####.###";A
50  PRINT USING "#####.###";B
60  PRINT USING "#####.###";C
70  END
```

このプログラムを RUN させると，……

```
␣␣␣18.500
␣␣␣␣2.670
␣␣135.780
```

……のように表示され，大変わかりやすくなります．

ここで用いられている "#" や "." がフォーマット文字列です．

同様に文字列の表示にも使うことができます．文字列のフォーマット文字列は & を使います．

```
10  AB$="CASIO␣COMPUTER":B$="!!"
20  PRINT USING "&&&&&&&&&&&&&&";AB$;B$;
30  END
```

このプログラムを RUN させると……

```
CASIO␣COMPUTER␣␣!!
```

……のように表示されます.

CASIO␣COMPUTERの文字列の数は14個ですが，フォーマット文字列&が16個書かれていますので，余分の2字分が空白となって表示されています.

USING文の使い方の注意を次に述べておきます.

(1) フォーマット文字列中には，"#"".""∧""&"以外の文字を書くことはできない.

(2) "#"".""∧"と"&"は混在することができない.

(3) 数値フォーマット指定

……数値の桁指定

. ……小数点指定

∧……指数部表示指定

① #の数は小数点より上位は13桁以内，下位は9桁以内とし，上位と下位を合わせて13桁以内で指定できる.

```
#############.
 (13個)
              .#########
               (9個)
#########.#####
 (13個)
```

② 負符号も#の中に含めて指定する.

③ ∧は末尾に書き，いくつ書かれていても同じ書式になる.

④ 小数部の桁数が，フォーマットより長い場合は，指定桁の次の数値を四捨五入して出力する.

PRINT USING "###.###";12.3456 ➡ 12.346

⑤ 整数部の桁数がフォーマットより長い場合は，先頭に%がついて，フォーマットに従わずに出力される.

PRINT USING "##.##";1234.56 ➡ %␣1234.56

⑥ 数値は右づめで出力される.

(4) 文字列フォーマット指定　&

① &の数はいくつ書いてもかまわない.

② &の数が文字列に対して少ない場合は，文字列の先頭から&の数だけ出力する.

PRINT USING "&&&";"ABCDEF" ➡ ABC

③ 文字列は左づめで出力される.

④ &の数が文字列より多い場合は，スペースを出力する.

(5) USING 指定は，1つの PRINT 文，もしくは LPRINT 文中のみで有効.

(6) USING 指定の変更は，新たな USING 指定により行なわれる.

(7) USING" "；により，USING 指定を解除することができる.

サンプル・プログラム

●文字列の各文字間にスペースをとって表示するプログラム

```
10 A$="CASIO"
20 PRINT A$
30 FOR I=1 TO LEN(A$)
40 M$=MID$(A$,I,1)
50 PRINT USING"&&&";M$;
60 NEXT I
70 PRINT
80 END
```

このプログラムを RUN すると……

```
CASIO
C␣␣A␣␣S␣␣I␣␣O
```

……のように表示されます.

MID$ 関数で，文字列を1文字ずつ取り出し，取り出した文字を，USING 関数のフォーマットにより表示させています.

サンプル・プログラム

●名前,身長,体重を一定のフォーマットでプリンターへ出力するプログラム

```
1100 REM ***USING***
1110 CLEAR :DIM A$(1),A!(1),B!(1)
```

```
 1120 FOR I=0 TO 1
 1130 INPUT "NAME";A$(I),"SINCHIYO (cm)"
;A!(I),"TAIJU(Kg)";B!(I)
 1140 LPRINT :NEXT I
 1150 FOR I=0 TO 1
 1160 LPRINT TAB(2);USING"&&&&&&&&&&&&&&
&";A$(I);
 1170 LPRINT USING"######";A!(I);"cm";B!
(I);"Kg"
 1180 NEXT I
 1190 END
```

実行例

```
ASASIO TAROU       190cm    185Kg
FUKAYA HIROKI       68cm      7Kg
```

表示用関数

POINT (point：ポイント)

機　能	ディスプレイ上のドットが点灯しているか否かを与える
書　式	POINT (x, y) 0 ≦ x ≦159 (水平位置) 0 ≦ y ≦31　(垂直位置)

表示されている画面を見ると，小さな正方形の点で，文字や記号などが構成されていることがわかります．

たとえば，Aというキャラクタは……

……となっています．

これらの1つ1つの点をドット（dot）と呼んでいます．

画面全体は，5120ドットで構成されています．

いま，横を x 軸，縦を y 軸とすると，画面上の1ドットは……

座標（x，y）

……で表わすことができます．

この（x，y）の座標で示されるドットが点灯しているか消えているかをチェックし，点灯していれば1，消えていれば0を与える関数が，POINT 関数です．

使い方

```
10  X=0:Y=0
20  A=POINT(X,Y)
30  PRINT  A
```

このプログラムをRUNすると，もし，（0，0）のドットが点灯していれば〝1〟を表示し，消えていれば〝0〟を表示します．

POINT関数の使用にあたっては，次の点を注意してください．

① xおよびyは，小数点第1位を四捨五入した値が用いられる．

② xおよびyが座標の範囲を超えた場合はBSエラーとなります．

サンプル・プログラム

●レーザ砲プログラム

SPC キーを押してレーザー砲を発射させ，動いている点を打ちます．

```
100 CLS
110 X=INT(RND*10)+75
120 DRAW(X,0):DRAW(80,29)-(80,31)
130 N$=INKEY$
140 IF N$=" " THEN GOSUB 300
150 DRAWC(X,0)
160 GOTO 110
300 DRAW(80,28)-(80,1)
310 A=POINT(80,0)
320 IF A=1 THEN LOCATE 0,3:PRINT "BEE!
":BEEP
330 DRAWC(80,28)-(80,1)
340 FOR I=0 TO 30:NEXT I
350 CLS :RETURN
```

数値関数

SIN (sine：サイン)

機　能	xの正弦　sin x を与える
書　式	SIN 数式　　　　−5400°＜数式＜5400°

SIN 関数は，xの正弦 sin xを計算します．

xは3つの角度の単位（DEG，RAD，GRA）を選んで使えます．

電源スイッチをONにした状態では，角度の単位はDEG(度)になっています．

使い方

DEG(度)を使って，SIN x の計算をしてみます．

```
10 PRINT SIN30
20 PRINT SIN45
30 PRINT SIN90
50 END
```

このプログラムをRUN（R U N ↵ あるいは SHIFT RUN/Q ↵）すると，SIN 30と SIN 45の結果が流れるように表示され，見えなくなってしまいます．

では，次の行を追加して，再びRUN してみてください．

```
25 STOP
```

結果は，SIN 30，SIN 45の結果が表示され，ストップします．

■実行例

```
 0.5
 0.7071067812
STOP  P0-25
_
```

続きが見たいときは，CONT↵ か SHIFT CONT/W ↵ と押すと，次の結果が表示されます．（参照　CONT 130ページ）

■実行例

```
 1
Ready  P0
_
```

次に，角度の単位（DEG，RAD，GRA）のいずれかを選んで，その後で，SIN x の計算を行なってみましょう．

```
10 REM SIN X  EXAMPLE
20 PRINT "ANGLE=";
30 INPUT K
40 ANGLE K
50 PRINT "SIN X:X=";
60 INPUT X
70 PRINT "SIN";X;"=";SINX
80 STOP
90 END
```

このプログラムを RUN すると，最初に……

ANGLE＝?

……と角度と単位を聞いてきます．

そこで，次のように，角度の単位を指定してやります．

ANGLE 0 ⟶ DEGREE （ディグリー：度）

ANGLE 1 ⟶ RADIAN （ラジアン）

ANGLE 2 ⟶ GRADIENT（グラディエント）

ためしに，ラジアンでやってみます．〝1〞を入力すると……

SIN X：X＝?

……と再び聞いてきます．そこで，たとえば，(PI/4)のようなラジアンの角度を入力すると，次のように結果が表示されます．

■実行例

```
SIN 0.7853981634= 0.
7071067812
STOP P0-80

_
```

以上のように，角度の単位を変更するときは，ANGLE コマンドを使いますが，各角度の単位に入力できる範囲は，次のように制限されています．

DEG　−5400°　<　数式　<　5400°
RAD　−30π　<　数式　<　30π
GRA　−6000　<　数式　<　6000

数式の値が上記の範囲を外れた場合はエラー（BS error）となります．

なお，引数には実数（たとえば，30など）の他に，変数や数式が使えます．

実数や変数1つのときには，引数をカッコで囲っても，囲わなくても，どちらでもよいのですが，数式のときには，カッコのあるなしで，結果が違ってきますので注意が必要です．たとえば……

SIN X+Y　……SIN Xの計算結果にYをたし算する
SIN (X+Y)　……X＋Yの計算結果のSINを計算する

……のようになります．

参照　ANGLE，COS，TAN

一口メモ

三角関数の角度の表わし方には，〝**度**（deg)〟，〝**ラジアン**（rad)，〝**グラディエント**(gra)〟の3種類があります．

度　°　………1°は円周の1／360
rad　………1 radは円周の1／2π
gra　………1 graは円周の1／400

主に使う角度は，度とラジアンですが，その関係は……

$$1^\circ = \pi / 180\text{rad} = 3.141592654 / 180\ \text{rad}$$

……です．

数値関数

COS (cosine：コサイン)

機　能	xの余弦 COS xを与える
書　式	COS 数式　　　　　－5400°＜数式＜5400°

COS関数は，xの余弦 COSx を計算します．

COS x の角度単位，引数xの入力範囲については，SIN x とまったく同じです．

使い方

COS x を使った，プログラムを入力してみましょう．

```
10 REM COSX EXAMPLE
20 PRINT "ANGLE=";
30 INPUT K
40 ANGLE K
50 PRINT "COSX:X=";
60 INPUT X
70 PRINT "COS";X;"=";COSX
80 STOP
90 END
```

このプログラムをRUNすると，次のように……

ANGLE＝?

……と角度の単位の入力を要求してきます．

そこでたとえば，グラジアンの単位で求めようとするときは“2”と入力します．すると，次のように角度を聞いてきますので，1355を入力してみます．

COS X : X= ?

結果は，－0.7604059656となります．

参照　SIN，ANGLE，TAN

数値関数

TAN (tangent：タンジェント)

機　　能	xの正接 TAN x を与える
書　　式	TAN 数式　　　　　−5400°＜数式＜5400°

TAN 関数は，x の正接 TAN x を計算します．

角度の単位については，SIN x，COS x と同じように扱います．

使い方

TAN x を求める，次のプログラムを入力してみましょう．

```
10 REM TANX EXAMPLE
20 PRINT "ANGLE=";
30 INPUT K
40 ANGLE K
50 PRINT "TANX:X=";
60 INPUT X
70 PRINT "TAN";X;"=";TANX
80 STOP
90 END
```

角度の単位を〝度〞にして（ANGLE 0 とする），TAN 45 を求めてみましょう．結果は，TAN 45＝1 と表示されます．

次に，TAN 90 を求めてみましょう．

今度は，〝MA error〞と表示されてしまいました．

TAN 関数は，90° に近づくにつれ，TAN x の値が急激に大きくなります．

そして，TAN 90 のときは無限大となり，数えられなくなってしまいます．

そのために，上記の例では，〝MA error〞が表示されたわけです．このように，TAN x においては，x が ±90°×(2n−1)（n は整数）の値をとるときには MA エラーとなりますので，注意してください．

参照　SIN，COS，ANGLE

数値関数

ASN, ACS, ATN

(arcsine ：アークサイン
arccosine：アークコサイン
arctangent：アークタンジェント)

機　能	ASN は 逆正弦 $\sin^{-1}$ x を与える ACS は 逆余弦 $\cos^{-1}$ x を与える ATN は 逆正接 $\tan^{-1}$ x を与える
書　式	ASN x, ACS x $\lvert x \rvert \leqq 1$: ATN x $\lvert x \rvert < 10^{100}$

ASN, ACS, ATN などは逆三角関数を表わし，それぞれ，逆正弦 $\sin^{-1} x$，逆余弦 $\cos^{-1} x$，逆正接 $\tan^{-1} x$ を計算します．

三角関数（sin, cos, tan）は，角度を与えて，その三角関数値を求める関数ですが，これらの逆三角関数は三角関数値を与えて，そのときの角度を求める関数です．

使い方

ASN x を使ったプログラム例を示します．

```
10 REM ASNX EXAMPLE
20 PRINT "ANGLE=";
30 INPUT K
40 ANGLE K
50 PRINT "ASNX:X=";
60 INPUT X
70 PRINT "ASN";X;"=";ASNX
80 STOP
90 END
```

このプログラムを RUN すると，次の２つの入力要求が表示されます．

ANGLE＝？　　0 ↵ ……“度”を指定する

ASN X：X=？　　1 ↵ ……三角関数値を入力する

それぞれに，上記のような値を入力すると……

ASN 1 ＝90

……と表示されます．

つまり，SIN x ＝1 の角度 x を求めたことになります．

前述のプログラムを，ACS，ATN などにかえて，ためしてみてください．

これらの逆三角関数で与えられる数の角度単位は，SIN，COS，TANなどと同様にANGLEで指定します．

計算結果として出てくる角度の出力範囲は，度（DEG）〔0〕の場合……

−90° ≦ ASN ≦ 90°

0° ≦ ACS ≦ 180°

−90° ≦ ATN ≦ 90°

……となります．

SIN x，COS x は，理論的に1を越えることがないので，ASN x，ACS x の引数 x の値は，1を越えない数値にする必要があります．

参照 SIN，COS，TAN，ANGLE

数値関数

SQR (square root：スクウェア・ルート)

機　能	平方根を与える
書　式	SQR 数式　　　　数式≧0

SQRは，平方根を求める関数で，$\sqrt{\ }$（**スクウェア・ルート**）に相当します．したがって……

$$SQR\ x = x\hat{}0.5 = \sqrt{x}$$

…となります．このとき，xの値は0より大きい値でなければなりません．

使い方

次のプログラムは，円の面積を入力してその円の半径を求めるものです．

```
10 REM SQRX EXAMPLE
20 PRINT "CIRCLE no MENSEKI=";
30 INPUT S
40 R=SQR(S/PI) …………PIの項参照
50 PRINT "CIRCLE no HANKEI=";R
60 LOCATE 0,2
70 END
```

このプログラムをRUNすると，……

CIRCLE no MENSEKI=?

……と聞いてきますので，たとえば，面積100を入れてみます．

すると，5.641895835と表示されます．

この値が，円の半径です．

では，もう一度RUNして，今度はマイナスの値を入れてみてください．

たちまち，MA errorが表示されます．SQR xの引数xがマイナスになると，SQR(S/PI)が虚数になってしまうからです．このようなエラーを避けるには，引数の正負をチェックするための次の行を追加するとよいでしょう．

```
35 IF S<0 THEN 20
```

数値関数

LOG, LGT (natural logarithm：ログ common logarithm：エルジーティー)

機　能	LOG x ……自然対数 $\log_e x$ を与える LGT x ……常用対数 $\log_{10} x$ を与える
書　式	LOG 数式 LGT 数式　　数式＞0

LOG x は，x の自然対数 $\log_e x$（別の表わし方では ln x）の値を計算します．この場合，〝e〟を自然対数の底（てい）と呼びます．

eは，次の値をとります．

e＝2.718281828……

また，LGT x は x の常用対数 $\log_{10}$ x の値を計算します．つまり，常用対数は，底が10である対数です．

使い方

次のプログラムは，x の値を順次与えることによって，LOG x を計算するものです．

```
10 REM LOGX EXAMPLE
20 PRINT "X=";
30 INPUT X
40 PRINT "LOG";X;"=";LOGX
50 GOTO 20
```

プログラムをRUNすると，……

X＝?

……と表示され，LOG x の x の値を要求してきます．

そこで，たとえば〝1〟と入力してみますと，LOG 1 ＝0と表示され，次の

の値を要求して来ます.

■実行例

[CLS]

RUN ↵

1 ↵

```
RUN
X=?
LOG  1=  0
X=?_
```

この対数関数 LOG x と指数関数 EXP x は, 次のグラフのように逆の関係にあります.

このグラフからもわかるとおり, 対数関数においては, x>0であることが必要です. もし負の値を入れると, MA error が表示されます.

LOG x は底が eである x の対数ですが, e 以外の底のときの対数 $\log_y x$ は, 次の式で計算できます.

LOG x ／LOG y

したがって, x の常用対数 LGT x は, 次の式でも求めることができます.

LOG x ／LOG 10

サンプル・プログラム

●底の値を入力し，さまざまな対数値を求めるプログラム

```
10 REM***LOGX/LOGY***
20 PRINT "X=";
30 INPUT X
40 PRINT "Y=";
50 INPUT Y
60 PRINT "LOG";X;"/LOG";Y;"=";LOG X/LO
   GY
70 LOCATE 0,3
75 STOP
80 END
```

このプログラムは，LOG x ／LOG y の式によって，$\log_y x$の値を計算するプログラムです．

実行例を，次に示します．

■実行例

RUN ↵
10 ↵
2 ↵

```
LOG 10/LOG 2= 3.
321928095
STOP P0-75
_
```

参照 EXP

数値関数

EXP (exponent：エクスポーネント)

機　　能	指数関数 e^x を与える
書　　式	EXP 数式　　　　　数式≦230

EXPは，exponentの略ですが，指数関数の値（e^x）を計算します．このとき，〝e〞を指数関数の底（てい）と呼びますが，eの値は……

$$e = 2.718281828\cdots\cdots\cdots$$

……です．

話のたとえ方として，「指数関数的に増大する」などと言いますが，この関数は引数xの値が大きくなると，急激にEXP xの値が大きくなるという性質をもっています．

使い方

では，次のプログラムを入力して，EXP xの値の変化を見てみましょう．

```
10 REM***EXP***
20 PRINT "A=";
30 INPUT A
40 FOR X=1 TO A
50 PRINT "EXP";X;"=";EXPX
60 FOR I=1 TO 300:NEXT I
70 NEXT X
80 END
```

このプログラムをRUNすると，EXP xの引数xの最大値を聞いてきます．

A＝?

そこで，たとえば，〝10〞と入力してみますと，⏎キーを押すごとに，次のページのような結果が表示されます．

■実行例

```
EXP 1= 2.718281828
EXP 2= 7.389056099
EXP 3= 20.08553692
EXP 4= 54.59815003
EXP 5= 148.4131591
EXP 6= 403.4287935
EXP 7= 1096.633158
EXP 8= 2980.957987
EXP 9= 8103.083928
EXP 10= 22026.46579
```

どうですか，EXP x の値の増えかたが急激であることがわかりますね．

さて，もう一度プログラムを走らせてみましょう．

今度は，A に〝231〟を入れてみてください．

延々とEXP x の値が表示され，引数 x の値が 231 になったとき，MA error が表示されました．

実は，EXP x の引数 x の入力範囲は，……

$x \leqq 230.2585$

……なのです．

x =230.2585 の EXP x の値は，次のようになります．

■実行例

```
EXP 230.2585= 9.999907006E99
```

この結果を見ますと，もう少しでオーバーフローしそうだということがおわかりになるでしょう．

なお，次の点にも注意してください．

(1)　| x | = 4E−10 に近くなると，EXP x =1　となる．

　x < −288 に近くなると，EXP x =0　となる．

数値関数

ABS (absolute：アブソリュート)

機　　能	引数の絶対値を与える
書　　式	ABS 数式

ABS xは，xの絶対値を与えます．数学的に表わすと……

$$ABS\ x = |x|$$

……となります．

ABS xのxが……

x≧0　（xの値が正の場合）……　ABS x＝x

x＜0　（xの値が負の場合）……　ABS x＝－x

……となります．

つまり，ABS x は，結果が正数（絶対値）となるように演算してくれるわけです．

使い方

ABS関数を使ったプログラムをためしてみましょう．

```
10 REM ABSX EXAMPLE
20 READ A,B,C,D
30 X=A:GOSUB 40:X=B:GOSUB 40:X=C:
   GOSUB 40:X=D:GOSUB 40:END
40 PRINT "ABS";X;"=";ABSX
50 FOR I=1 TO 200:NEXT I
60 RETURN
70 DATA5,-5,0,-7.5
```

このプログラムはREAD文で，変数A，B，C，Dに，5，－5，0，－7.5を読み込み，それぞれ，ABS A ～ABS Dを計算します．

結果は次のように表示されます．

■実行例

```
ABS 5=5
ABS-5=5
ABS 0=0
ABS-7.5=7.5
```

なお，次のようにSGN関数を使って，ABS関数と同じ働きをさせることができます．

ABS x ⟵ 等しい ⟶ x＊SGN x

サンプル・プログラム

●負の値を入力しても，エラーの生じないプログラム

```
10 INPUT X
20 S=SQR(ABSX)
30 L=LOG(ABSX)
40 PRINT "SQRX=";S
50 FOR I=1 TO 200:NEXT I
60 PRINT "LOGX=";L
70 END
```

SQR x やLOG x などの関数は，x に負の値が入力されると，MA error となります．

そこで，このプログラムではABS関数で x の絶対値をとり，計算するようにしてあります．

参照　SGN

数値関数

INT (integer：インテジャー)

機能	引数の値を超えない最大の整数を与える
書式	INT 数式

INT x は，xの値を超えない最大の整数値を与えます．

たとえば，xの値が 3.9，0.5，−0.5，−3.9 などのとき，INT x は，それぞれ次のようになります．

INT 3.9 ＝3

INT 0.5 ＝0

INT −0.5 ＝−1

INT −3.9 ＝−4

引数の値が，正のときは，単純に小数点以下を切り捨てた値となりますが，負の場合は，注意が必要です．たとえば，−0.5の場合は0とはならず，−0.5を超えない最大の整数値ですから，−1となるわけです．

使い方

では，次のプログラムでためしてみましょう．

```
10 REM INTX EXAMPLE
20 READ A,B,C
30 X=A:GOSUB 40:X=B:GOSUB
   40:X=C:GOSUB 40:END
40 PRINT "INT";X;"=";INTX
50 FOR I=1 TO 200:NEXT I
60 RETURN
70 DATA5.3,0.5,-3.9
```

■実行例

```
INT 5.3= 5
INT 0.5= 0
INT-3.9=-4
```

プログラムをRUNすると，前ページのような結果を表示します．

xの値を横軸にとり，INT xの値を縦軸にとって，INT関数のグラフを描くと，次のようになります．

INT xのグラフ

このグラフによって，正負のとる値の違いがおわかりになるでしょう．

なお，INT関数は他の関数，たとえばRND関数などと組み合せてよく使われます．また，INT関数に類似した関数として，FRAC関数やROUND関数などがあります．

サンプル・プログラム

●0～9の整数をランダムに5つ表示するプログラム

```
10 FOR I=1 TO 5
20 PRINT INT(10*RND);
30 NEXT I
40 END
```

このプログラムは，INT関数とRND関数を組み合せて使った例です．

参照　FRAC，ROUND，RND

数値関数

FRAC (fraction：フラクション)

機　　能	引数の小数部を与える
書　　式	FRAC 数式

FRAC x は，x の小数部分を与えます．
簡単な例を示しておきましょう．

FRAC 1.123 = 0.123

FRAC −1.123 = −0.123

このように，単純に整数部を切り捨てる働きをします．

使い方

次のプログラムに，いろいろな値を入力してためしてみてください．

```
10 REM FRACX EXAMPLE
20 PRINT "NUMBER";
30 INPUT X
40 PRINT FRACX
50 GOTO 10
```

サンプル・プログラム

● RND 関数で発生した小数点以下 9 桁の乱数を，1 桁ずつの整数として取り出すプログラム

```
10 DIM Y(9)
20 X=RND
30 FOR I=1 TO 9
40 Y(I)=INT(10*X)
50 PRINT "X=";X
60 PRINT "Y(";I;")=";Y(I)
70 FOR J=1 TO 150:NEXT J
80 X=FRAC(10*X)
90 NEXT I
```

このプログラムは，最初にRND関数で発生した値を変数Xに代入します．

その値を10倍し，INT関数で1桁の整数を得て，配列変数Y(1)に代入し，残りの小数点以下をFRAC関数で取り出し，Xに代入し，新たなXの値とします．これを9回くり返し，配列変数Y(1)～Y(9)に，1桁ずつの乱数を分配します．

たとえば，最初に，次のような乱数が発生したとします．

……のように，分配されます．

参照　INT，RND，ROUND

数値関数

SGN (sign：サイン)

機　能	引数の符号（正負）を与える
書　式	SGN 数式

SGN x は，引数 x の値が，正であるか負であるかを判別する働きをします．SGN x の結果は，次の 3 種類を与えます．

x >0　（正の場合）…………　SGN x ＝ 1
x =0　（0 の場合）…………　SGN x ＝ 0
x <0　（負の場合）…………　SGN x ＝－1

使い方

次のプログラムで，ためしてください．

```
10 REM SGNX EXAMPLE
20 PRINT "+ - no HANTEI"
30 PRINT "KAZU";
40 INPUT X
50 A=SGNX
60 IF A=1 THEN PRINT "+":GOTO 30
70 IF A=0 THEN PRINT "0":GOTO 30
80 IF A=-1 THEN PRINT "-":GOTO 30
```

60～70行において，SGN x で得た値をもとに，正なら〝+″，負なら〝−″，0 なら〝0″を表示させるようにしています．

サンプル・プログラム

●sin カーブを描くプログラム

```
10 CLS :FOR X=0 TO 540 STEP 20
20 S=SGN(SINX)
30 Y=S*INT(S*10*SINX)
```

```
40 IF S<0 THEN 70
50 DRAW(X/4,16-Y)
60 GOTO 80
70 DRAW(X/4,16+Y)
80 NEXT X
90 END
```

このプログラムは，SIN X のXを0～540°まで20°増しに求め，その値を10倍して1桁の値として，Yに代入します．

そのとき，SIN X の SGN 関数をとり，SIN X が負のときでも，INT 関数で整数部のみが取り出せるようにしています．

このようにして求めたX，Yの値を，ドット座標（0,16）－（159,16）を中心に，DRAW コマンドで正のときは（X/4，16—Y），負のときは（X/4,16+Y）の座標にドットを打たせています．
（DRAW の項を参照してください.）

このプログラムを RUN すると，次のような粗いサインカーブが描かれます．

■実行結果

数値関数

ROUND (round：ラウンド)

機　能	数式１の値を，10の「数式２の値」乗の桁で四捨五入します
書　式	ROUND（数式１，数式２）

ROUND(x, y)は，xの値を10^{y}の桁で四捨五入した値を与えます．

たとえば……

ROUND(12345, 2) = 12000

……となります．数値12345を10^{2}，つまり100の位で四捨五入しているわけです．

使い方

次のプログラムで，ROUND関数の働きを確認していただきましょう．

```
10 FOR Y=3 TO -5 STEP -1
20 X=12345.67891
30 Z=ROUND(X,Y)
40 PRINT USING"#####.#####";X,Z:PRINT
50 FOR J=0 TO 150:NEXT J
60 NEXT Y
70 END
```

このプログラムは，ROUND(x, y)のyの値を3～−5まで１ずつ減らして計算したものですが，次のページの結果のように，10^{y}の位で四捨五入されているのがおわかりになるでしょう．

なお，第２番目の引数yに指定できる値は……

$|y| < 100$

……と決められています．これ以上では，BS errorとなります．

また，yの値に小数以下をもつ値を指定したときは，小数部を切り捨てた位置で桁指定します．

■実行例

```
12345.67891
10000.00000
12345.67891
12000.00000
12345.67891
12300.00000
12345.67891
12350.00000
12345.67891
```

サンプル・プログラム

● 2進数8ビットの乱数の発生と，その10進数の値も表示するプログラム

```
 10 Y=0
 20 FOR I=7 TO 0 STEP -1
 30 X=ROUND(RND,-1)
 40 Y=Y+(2^I)*X
 50 PRINT X;
 60 NEXT I
 70 PRINT "=";Y
 80 LOCATE 0,1
 90 FOR J=0 TO 500:NEXT J
100 END
```

30行で，RND関数(259ページ参照)で発生した小数点以下の値を，ROUND関数で小数点以下第1位で四捨五入していますので，Xの値は0か1のいずれかの値となります．このXの値を8回発生させて，0と1で構成された8ビットの乱数を作ります．

同時に，40行で2進数8ビットの値を10進数に変換し，その値も連続して表示させます．実行例を次に示しておきます．

RUN ↵

```
RUN
0 0 1 0 1 0 0 1= 41
```

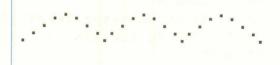
数値関数

PI (pi：パイ)

機　能	円周率 π を与える
書　式	PI

PI は円周率 π の概数を与えます.

与えられる PI の値は，11桁で次のとおりです.

$$\pi = 3.1415926536\cdots\cdots$$

(なお，この値は内部演算上，取られる値なので，表示上は，PI=3.141592654となります.)

使い方

次のプログラムは，円の面積を計算するものです.

```
10 REM PI EXAMPLE
20 PRINT "EN no MENSEKI"
30 PRINT "HANKEI WA";
40 INPUT R
50 S=PI*R^2
60 PRINT "S=";S
70 END
```

たとえば，半径の値として5を入力しますと，次のように円の面積を表示します.

■実行例

```
S= 78.53981634
```

数値関数

RND (random number：ランダム・ナンバー)

機　　能	乱数値を与える
書　　式	RND　　　　0<RND<1

RND 関数は，0より大きく1未満の10桁の疑似乱数値を与えます．

乱数とは，その発生する前後になんの規則性もない，つまり次にくる値があらかじめ予測できない数をいいます．

もともと，統計現象，確率モデルのシミュレーション用として必要であった乱数が，最近は，経済予測などのシミュレーションや，テレビゲームなどに応用されています．テレビゲームの面白さは，この乱数関数によっているところが大きいのです．

使い方

次のプログラムは，10個の乱数を発生させ，表示するものです．

```
10 FOR N=1 TO 10
20 PRINT RND
30 FOR X=1 TO 500:NEXT X
40 NEXT N
50 END
```

結果は，次のようになりましたが，当然あなたがRUNした場合には，異なる乱数が発生します．

これを見てもわかるように，全く規則性がないことがわかります．

```
0.6791506196
0.959823211
0.205719988
0.503905755
0.306977109
0.106577855
```

```
0.4177075471
0.501741468
0.755155195
0.456091832
```

なお，このままの値では桁数が多く扱いにくいので，ゲームなどに応用するときは，次のように，INT関数やROUND関数と組み合せて，適当な範囲の乱数値として使います．

① 希望する桁までの整数を取り出す

INT(RND＊10^L)………Lは桁数

② Nから上限Mまでの整数を取り出す

ROUND(RND＊(M−N)，−1)+N……N，Mは整数（N<M）

第5章 プログラムライブラリー

プログラムに関するお問合せは，下記へ葉書でお願いします．
〒102 東京都千代田区平河町1-4-12
㈱技術評論社
ソフトウェア担当
●なお，本書のプログラムの利用によって生じるいかなる結果に対しても，カシオ計算機㈱及び㈱技術評論社は一切の責任を負うことはできません．

別売品のカセットインタフェース付，ミニプロッタプリンタを付けないで，本章のプログラムを実行する場合，PRINTER ON？(Y/N)で，プリントアウトするかどうかを聞いてきたときは，Nを押して実行を続けて下さい．Yを押すと，NRエラーが出ます．

株価管理と適正売・買値

過去53週（つまり一年分）の株価を記憶します．53週以後は，古い株価から順に捨てていきます．このデータをもとに，現在の偏差値を出したり，売り，買いの判断材料を出力します．また，偏差値と移動平均をかみ合わせてみることや，グラフィック画面を利用して，株価の変動をグラフにしてみることもできます．

解　説

プログラムはＰ０からスタートさせます．すると画面にメニューが表示され，番号の入力を要求してきますので，メニューの後についている番号を入力して下さい．範囲は１～７です．これ以外の数字を入力すると，もう一度メニューの表示になります．

メニューで１を押すと，データが入力されます．"HAJIME(Y/N)"かきいてきますから，Ｙを押しますと，"CLEAR OK?(Y/N)" と画面に出ます．これは入力ミスによるデータ消失を防ぐためのものです．普通はＹを入力します．

次にデータをきいてきますから，数字を入力して ⏎ キーを押します．２つ目からのデータの入力は，何週目かは" シュウ=" で示されます．53週目以降は，前のデータを１つずつずらしていくので，入力に少し時間がかかります．

２は売り買いの判断です．

３は，適正価格のチェックです．このルーチンでは，新たにデータを入れなおした場合，２のルーチンを実行した後でなければ正しい値を表示しないので注意して下さい．これは，変数の一部を共用しているためです．そのかわり，このルーチンでは入力に待たれることはありません．

なお，１から３のそれぞれのルーチンから抜けたいときは，負の数を入力すれば，メニューに戻ります．

４でデータの出力をし，５で移動平均を求めます．

６は過去の移動平均を見ることによって，株価の上がり下がりいずれかの基調を調べます．

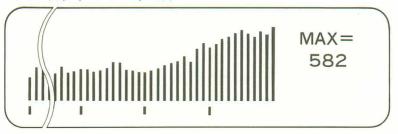
また，５と６とでは，特にデータが少ない時の入力に気をつけて下さい．まちがった値の場合，エラーがでることがあります．

７は，全体の流れが見やすいようにグラフを作成するものです．下の目盛りは10週おきに出ていますが，データが53週分より少ない時は気をつけて下さい．

なお，このプログラムのDATA文および，サブルーチンは，すべてカナなどを出力するためのものです．

グラフ表示（メニュー7）の例

プログラム

P0

```
 10 CLS :LOCATE 0,0:PRINT "INPUT 1 ":G
    OSUB 150
 20 FOR K=0 TO 50:NEXT K
 30 GOSUB 200
 40 FOR K=0 TO 100:NEXT K
 50 LOCATE 9,2:PRINT "GRAPH 7"
 60 LOCATE 10,3:INPUT "NO.";PR
 70 IF PR>7 THEN 10 ELSE IF PR<1 THEN
    10
 80 IF PR=1 THEN GOTO PROG 1 ELSE IF P
    R=2 THEN GOTO PROG 2
 90 IF PR=3 THEN GOTO PROG 3 ELSE IF P
    R=4 THEN GOTO PROG 4
100 IF PR=5 THEN GOTO PROG 5 ELSE IF P
    R=6 THEN GOTO PROG 6 ELSE GOTO PRO
    G 7
150 RESTORE :X=4
160 X=X+1:FOR K=1 TO X
170 READ DA:PRINT CHR$(DA);:NEXT K:PRI
    NT X-3:IF X=5 THEN 160
180 RETURN
200 FOR X=1 TO 5:READ DA:PRINT CHR$(DA
    );:NEXT X:PRINT 4;
210 X=4
220 X=X+1:LOCATE 9,X-5:FOR K=1 TO 9:RE
    AD DA:PRINT CHR$(DA);:NEXT K:PRINT
     X
230 IF X=5 THEN 220
240 RETURN
```

```
400 DATA182,204,222,32,182
410 DATA195,183,190,178,32,182
420 DATA203,174,179,188,222
430 DATA178,196,222,179,32,205,178,183
    ,221
440 DATA182,186,32,201,32,178,196,222,
    179
```

P1

```
 10 INPUT "HAJIME(Y/N)",P$:IF P$="Y" T
    HEN GOSUB 200
 20 IF P$="N" THEN A=A-1 ELSE IF P$<>"
    Y" THEN 10
 30 GOSUB 300
 40 A=A+1:GOSUB 330:PRINT A,
 50 IF A>=53 THEN 70
 60 INPUT "DATA",Z(A):IF Z(A)<0 THEN Z
    (A)=0:GOTO PROG 0 ELSE GOTO 40
 70 INPUT "DATA+",DZ:C=C+1:A=53:IF DZ<
    0 THEN GOTO PROG 0 ELSE Z(53)=DZ
 80 FOR B=1 TO 53
 90 Z(B-1)=Z(B)
100 NEXT B
110 GOTO 70
200 INPUT "CLEAR OK?(Y/N)",T$:IF T$="Y
    " THEN 220 ELSE IF T$="N" THEN 240
210 GOTO 200
220 ERASE Z :DIM Z(53)
230 A=0
240 P$="Y":RETURN
300 RESTORE :FOR X=1 TO 5
310 GOSUB 350:NEXT X
320 PRINT " DATA":RETURN
330 RESTORE 460:FOR K=1 TO 4:GOSUB 350
    :NEXT K:RETURN
350 READ DA:PRINT CHR$(DA);:RETURN
450 DATA182,204,222,32,182
460 DATA188,173,179,61
```

P2

```
 10 GOSUB 200:PRINT " CHECK"
 20 S=0:Q=0
 30 FOR D=0 TO 52
 40 S=S+Z(D):Q=Q+Z(D)^2
 50 NEXT D
 60 IF A<54 THEN 80
 70 E=S/53:V=Q-53*E*E:F=SQR(V/52)
```

```
 80 E=S/A:V=Q-A*E*E:F=SQR(V/(A-1))
 90 GOSUB 200:INPUT "=",Y:IF Y<0 THEN
    130
100 H=ROUND(50+10*(Y-E)/F,-3)
110 GOSUB 220:PRINT H
120 GOTO 90
130 GOTO PROG 0
200 RESTORE :FOR K=1 TO 4:READ DA
210 PRINT CHR$(DA);:NEXT K:RETURN
220 RESTORE 310:FOR K=1 TO 5:READ DA
230 PRINT CHR$(DA);:NEXT K:RETURN
300 DATA182,204,222,182
310 DATA205,221,187,193,61
P3
 10 INPUT "H=",H:Y=ROUND((H-50)*F/10+E
    ,-2):IF H<0 THEN 30
 20 GOSUB 100:PRINT Y:GOTO 10
 30 GOTO PROG 0
100 RESTORE :FOR K=1 TO 5
110 READ DA:PRINT CHR$(DA);:NEXT K
120 RETURN
200 DATA182,204,222,182,61
P4
 10 FOR U=0 TO A-2
 20 PRINT "DATA";U+1;"=";Z(U+1)
 30 FOR K=0 TO 50
 40 NEXT K:NEXT U
 50 PRINT "DATA END"
 60 FOR K=1 TO 200:NEXT K
 70 GOTO PROG 0
P5
 10 X=0
 20 GOSUB 200:INPUT N
 30 IF A<=53 THEN 80
 40 FOR L=53-N TO 52
 50 X=X+Z(L)
 60 NEXT L
 70 GOTO 110
 80 FOR K=A-N TO A
 90 X=X+Z(K)
100 NEXT K
110 M=X/N
120 GOSUB 240:PRINT USING"####.###";M
130 FOR K=0 TO 300:NEXT K
140 GOTO PROG 0
200 RESTORE :FOR K=1 TO 6
210 READ DA:PRINT CHR$(DA);:NEXT K
230 RETURN
```

```
240 RESTORE 310:FOR K=1 TO 9
250 READ DA:PRINT CHR$(DA);:NEXT K
260 RETURN
300 DATA178,196,222,179,189,179
310 DATA178,196,222,179,32,205,178,183
    ,221
```
P6
```
 10 GOSUB 200:INPUT I
 20 GOSUB 220:INPUT O
 30 X=0
 40 IF A<=53 THEN 90
 50 FOR J=53-O-I TO A-O-I
 60 X=X+Z(J)
 70 NEXT J
 80 GOTO 120
 90 FOR J=A-O-I TO A-O-1:IF A<=J THEN
    140
100 X=X+Z(J)
110 NEXT J
120 M=X/I
130 GOSUB 240:PRINT M:O=O-1:X=0:FOR K=
    0 TO 20:NEXT K:GOTO 40
140 GOSUB 270:PRINT " END"
150 FOR K=0 TO 200:NEXT K
160 GOTO PROG 0
200 RESTORE :FOR K=1 TO 5
210 GOSUB 290:NEXT K:FOR K=1 TO 4:GOSU
    B 290:NEXT K:RETURN
220 RESTORE 320:FOR K=1 TO 8:GOSUB 290
230 NEXT K:RETURN
240 RESTORE :FOR K=1 TO 4:GOSUB 290:NE
    XT K
250 RESTORE 330:FOR K=1 TO 4:GOSUB 290
260 NEXT K:RETURN
270 RESTORE 330:FOR K=1 TO 4:GOSUB 290
280 NEXT K:RETURN
290 READ DA:PRINT CHR$(DA);:RETURN
300 DATA182,186,32,201,32
310 DATA178,196,222,179
320 DATA197,221,188,173,179,32,207,180
330 DATA205,178,183,221
```
P7
```
 10 CLS
 20 MX=0:FOR MD=1 TO 53
```

```
30 IF Z(MD)>MX THEN MX=Z(MD)
40 NEXT MD
50 LOCATE 15,2:PRINT "MAX=":LOCATE 15
   ,3:PRINT MX
55 FOR K=1 TO A
60 J1=K*2+10:J2=25-25/MX*Z(K)
70 FOR K=1 TO 5:PO=53*2+10-K*20+2
80 DRAW(PO,27)-(PO,30):NEXT K
90 FOR K=1 TO A
100 J1=K*2+10:J2=25-25/MX*Z(K)
110 DRAW(J1,J2)-(J1,25):NEXT K
120 IF INKEY$="" THEN 120
130 GOTO PROG 0
```

テ　ス　ト　デ　ー　タ							
19週前	584	14週前	545	9週前	635	4週前	685
18週前	580	13週前	550	8週前	652	3週前	697
17週前	579	12週前	563	7週前	673	2週前	685
16週前	570	11週前	589	6週前	701	1週前	672
15週前	562	10週前	620	5週前	692	今　週	689

メ　モ　リ　ー　内　容					
A	データ数カウンタ	J1	グラフX軸	Q	データ二乗和
B～D	カウンタ	J2	グラフY軸	S	データ合計
DA	文字データ	MD	カウンタ	T$	CLEAR OK?
DZ	株価データ	MX	最大のデータ	U	カウンタ
E	平　均	N	移動数	V	分　散
H	偏差値	O	移動平均始め週	X	カウンタ
J～L	カウンタ（P5においては移動平均区間）	P$	YorN（初めか？）	Z(X)	株価データ
		PØ	グラフスケール		
		PR	プログラム選択		

操　作

	手順	キー操作	表　示
メニュー →		SHIFT PØ	INPUT 1　イドウヘイキン 5 カブカ 2　カコノイドウ 6 テキセイカ 3　GRAPH 7 ヒョウジ 4　NO. ?
データ入力を指定→	1	1 ↵	HAJIME (Y/N)
初めてのときはY, 追加のときはN →	2	Y ↵	CLEAR OK ? (Y/N)
データをクリアしてもよければY, 否ならばN	3	Y ↵	カブカ DATA
1データ入力	4	584 ↵ ⋮	シュウ=1 DATA シュウ=2 DATA ⋮

	手順	キー操作	表示
負の数を入力するとデータ入力が終わる→	5	−1 ↵	∫ NO. ?
メニュー―――――→ [2]株価チェック	6	2 ↵	カブカCHECK カブカ＝
現在の株価を入力→	7	585 ↵	ヘンサチ＝49.23 カブカ＝
過去のデータにもとづく偏差値が出力される． 負の数の入力でメニューに戻る．――→	8	−1 ↵	∫ NO. ?
[3]適正価格	9	3 ↵	H＝
偏差値を入力――→ 株価が出力される→	10	55 ↵	カブカ＝670.1 H＝
負の数の入力でメニューに戻る――→	11	−1 ↵	∫ NO. ?
[4]データの 出力 全部のデータを出力してメニューに戻る→ メニュー―――――→	12	4 ↵	DATA 1＝584 DATA 2＝580 DATA 3＝579 ⋮ DATA END ∫ NO. ?
[5]移動平均	13	5 ↵	イドウ スウ？
移動数を入力――→	14	15 ↵	イドウ ヘイキン 643.200 ∫ NO. ?
[6]過去における移動平均	15	6 ↵	カコ ノ イドウ？
何週分の移動平均をとるかを入力――→	16	3 ↵	ナンシュウ マエ？
何週前からとるかを入力――――→ 出力が終わると――→メニューに戻る	17	2 ↵	カコ ノ ヘイキン 689 カコ ノ ヘイキン 684.666………… カコ ノ ヘイキン 682 ∫ NO. ?
[7]グラフ	18	7 ↵	グラフを作成

電　話　帳

名前と電話番号を登録しておけば，いつでもどこでも，す速くかけたい人の電話番号がわかります．名前の頭文字だけで調べたり，名前をアルファベット順に並べ変える等の機能を備えています．

解　説

友だちや知り合いの電話番号をこのプログラムに登録しておけば，電話したいときに，すぐ電話番号を知ることができます．一度に登録できる人数は50人までですが，一旦登録しておけば名前の頭文字や最初の数文字を入力するだけで電話番号を検索することができます．

まずＰ０をRUNして下さい．メニューが表示されます．１～４の数字で入力して下さい．

1—INPUT……名前と電話番号を登録します．下記の要領で入力して下さい．下線部分が打ち込み箇所です．

NAME　　? CASIO ↵　　　　（名前は，16文字まで）
TEL NO. ? 03－347－4830 ↵　（電話番号は，12文字まで）

以上の繰り返しで登録を行ないます．なお登録を終了したら，名前の代わりに「END」を入力して下さい．メニューに戻ります．

2—SORTING……名前をアルファベット順に並べ変えます．並べ変えている間は「SORTING...」と表示されます．登録量により異なりますが，数秒から数分で終了し，並べ変えた後の順序で登録されている名前と電話番号を１人ずつ表示します．キー（何のキーでも良い）を押すたびに次の人の表示に移り，全員表示し終えるとメニューに戻ります．

3—LOOK FOR……名前を入力して電話番号を検索します．下記の要領で名前を入力すると，名前と電話番号が表示されます．

NAME　　? CASIO ↵
CASIO
03－347－4830

名前の入力に際しては，頭文字や最初の数文字のみでも検索可能です．この場合，相当する名前のものはすべて表示します．また同じ名前が登録されている場合も，すべて表示するようになっています．

登録されていない名前を検索した場合は「NO DATA」を表示します．

何かのキーを押してメニューへ戻って下さい.

4—DELETE……登録済みのデータを削除する場合に使います. この場合も今までと同様に名前を入力します.

NAME　　?ABCDE ↵

ABCDE

000-0000 Y/N?

ここで, 本当に消す場合は「Y」を, 消したくない場合は「N」を押して下さい. これで, あやまって消してしまうことはないでしょう.

なお, 新たに登録しなおしたい場合は「CLEAR」を実行して下さい. 今までのデータはすべてクリアされます,. また, 1—INPUTは追加登録を行なう働きをしますから, 4—DELETEと組み合わせて, 自由にデータの追加, 削除が行なえます.

プログラム

```
P0
  10 CLS
  20 PRINT "1-INPUT   2-SORTING"
  30 PRINT "3-LOOK FOR"
  40 PRINT "4-DELETE"
  50 K$=INKEY$:IF K$="" THEN 50
  60 IF K$="1" THEN GOTO PROG 1
  70 IF K$="2" THEN GOTO PROG 2
  80 IF K$="3" THEN GOTO PROG 3
  90 IF K$="4" THEN GOTO PROG 4
 100 GOTO 50
P1
  10 CLS
  20 N=N+1
  30 IF N=51 THEN 90
  40 IF N=1 THEN DIM A$(50),B$(50)*12
  50 INPUT "NAME    ";A$(N)
  60 IF A$(N)="END" THEN 100
  70 INPUT "TEL NO.";B$(N)
  80 GOTO 10
  90 PRINT "FULL":BEEP 1
 100 N=N-1
 110 GOTO PROG 0
```

P2

```
  5 CLS :PRINT "SORTING."
 10 FOR I=1 TO N
 20 MM$=A$(I):X=I
 30 FOR J=I TO N
 40 KK$=A$(J)
 50 GOSUB 200
 60 NEXT J
 70 A$(X)=A$(I):A$(I)=MM$
 80 MM$=B$(X)
 90 B$(X)=B$(I):B$(I)=MM$
100 NEXT I
110 GOTO PROG 5
200 KU=0
210 KU=KU+1
230 O1=LEN(MM$):O2=LEN(KK$)
240 IF KU>O1 THEN RETURN  ELSE IF KU>O
    2 THEN 300
250 MI$=MID$(MM$,KU,1):KI$=MID$(KK$,KU
    ,1)
260 IF ASC(MI$)=ASC(KI$) THEN 210
270 IF ASC(MI$)>ASC(KI$) THEN X=J:MM$=
    KK$:RETURN
280 RETURN
300 X=J:MM$=KK$:RETURN
```

P3

```
 10 CLS
 20 INPUT "NAME  ";MM$
 30 X=LEN(MM$)
 40 I=0
 50 I=I+1
 60 IF I=N+1 THEN IF F=1 THEN 120 ELSE
     130
 70 IF MM$=LEFT$(A$(I),X) THEN 90
 80 GOTO 50
 90 F=1:PRINT A$(I)
100 PRINT B$(I)
110 K$=INKEY$:IF K$="" THEN 110 ELSE 5
    0
```

```
 120 F=0:GOTO PROG 0
 130 PRINT "NO DATA"
 140 K$=INKEY$:IF K$="" THEN 140 ELSE 1
     20
P4
  10 CLS
  20 INPUT "NAME  ";MM$
  30 X=LEN(MM$)
  40 I=0
  50 I=I+1
  60 IF I=N+1 THEN 180
  70 IF MM$=LEFT$(A$(I),X) THEN 100
  90 GOTO 50
 100 PRINT A$(I)
 110 PRINT B$(I);" Y/N ?"
 120 K$=INKEY$:IF K$="" THEN 120
 130 IF K$="Y" THEN 150
 140 GOTO 50
 150 A$(I)=A$(N)
 160 B$(I)=B$(N)
 170 N=N-1
 180 GOTO PROG 0
P5
  10 FOR I=1 TO N
  20 CLS
  30 PRINT A$(I)
  40 PRINT B$(I)
  50 K$=INKEY$:IF K$="" THEN 50
  60 NEXT I
  70 GOTO PROG 0
```

操作

	手順	キー操作	表示
メニューの表示→		SHIFT P0/Ø	1―INPUT 2―SORTING 3―LOOK FOR 4―DELETE
登録 メニューの1を選択→	1	1	NAME ?
最初の1人の登録(氏名)→	2	HARADA TOMOYUKI ↵	NAME ? HARADA TOMOYUKI TEL NO. ?
番号の登録→	3	03-583-4111 ↵	NAME ?
2人目の氏名の登録→	4	NISHI YUUZO ↵	NAME ? NISHI YUUZO TEL NO. ?
番号の登録→	5	052-264-1453 ↵	NAME ? ⋮
終了したら,「END」を入力. メニュー画面に戻ります.→	6	END ↵	1―INPUT 2―SORTING 3―LOOK FOR 4―DELETE
検索 だれをさがしますか?→	7	3	NAME ?
姓だけでもOKです.→	8	HARADA TOMOYUKI ↵	NAME ? HARADA TOMOYUKI HARADA TOMOYUKI 03-583-4111
メニュー画面に戻します.→	9	↵	1―INPUT 2―SORTNG 3―LOOK FOR 4―DELETE
並べかえ アルファベット順→	10	2	SORING...
「SORTING……」が消えたら→	11	↵	TAHARA YUUKO 06-314-2681
並べかえの結果を次つぎと表示→	12	↵	HARADA TOMOYUKI 03-583-4111 ⋮

	手順	キー操作	表示
並べかえの表示が終ると，メニュー画面に戻ります．→	13	↵	1－INPUT　2－SORTING 3－LOOK FOR 4－DELETE
削除 削除したい名前は？→	14	4	NAME　?
削除してよければ，Y，否ならばN →	15	HARADA TOMOYUKI ↵	NAME　? HARADA TOMOYUKI HARADA TOMOYUKI 03-583-4111 Y／N ?
メニューに戻ります．→	16	Y	1－INPUT　2－SORTING 3－LOOK FOR 4－DELETE
検索で確認してみます．→	17	3	NAME　?
「NO DATA」の表示は，該当なしを表します．→	18	HARADA TOMOYUKI ↵	NAME　? HARADA TOMOYUKI NO DATA

メモリー内容	
N	NAME，TEL No.の個数－1
X	捜す文字列の長さ
A$(0,0) ～ A$(50,0)	NAME
A$(0,1) ～ A$(50,1)	TEL No.
MM$	捜す文字列

CROSS TOTAL

よこ(X)とたて(Y)の2つの項目について，各項目の和を求めたり，ソーティング（並びかえ）を行なって，構成比を見たりするプログラムです．例えば，よこ軸に製品，たて軸に月をとれば，製品合計や，月別統計などのデータを得ることができます．

使い方

Ｐ０でプログラムをRUNさせます．まずプリンターを使用するかの確認後，次のメニューが表示されます．

```
1  DATA INPUT          ← データの入力をしたいときは1
2  TOTAL               ← よこ，たての合計を得たいときは2
3  SORT                ← 並べかえをしたいときは3
4  DATA OUTPUT ?_      ← 全データを見たいときは4
```

最初にデータを入力するときは，１，よこ，たての小計などを見たいときは，２，データの大きい順に項目を並べて構成比を見たいときには，３，今まで入れたデータを見たいときには，４を入力してください．

1. データの入力法（＊配列の取り方によって，入力数が異なります．）

メニューで，１を選択すると，「CLEAR(Y／N)?」と表示が出ます．初めてデータを入力するときは，ここで“Ｙ”を入力して下さい．

以後，Ｘ(よこの項目数)，Ｙ(たての項目数)をそれぞれいくつとるかを聞いてきますから，それぞれの数を入力して下さい．

次に，「INPUT DATA X= 1 Y= 1」で，Ｘ(よこ)，Ｙ(たて)のデータを聞いてきますから，数値を入力します．Ｘ＝とＹ＝のあとの数字は，次つぎに変化し，データは，よこの方向へ入れていきます．

X=1, Y=1 → X=2, Y=1 → X=3, Y=1…

X=1, Y=2 → X=2, Y=2 → X=3, Y=2…

	X1→	2	3	4………
Y1				
2				
3				
4				
⋮				

すべてのデータの入力が終わると，総合計(GRAND TOTAL)を表示して，“PRINTER ON?(Y/N)”の表示となります．

ＹかＮの入力によって，メニューに戻ります．

2. データの訂正

データの訂正をしたいときは，メニュー画面で，1を選択し，次の表示“CLEAR(Y/N)?”でNを入力すると，“TEISEI(X,Y)X=?”の表示で，訂正した箇所が，X(よこ)の何番目かを聞いてきます．Xの値を入力すると，次にY(たて)の何番目かを聞いてきます．ここで，Yの値を入力すると，その場所のデータを表示し，“DATA?”で新しいデータの入力を求めてきます．ここで，修正したい数値を入力すると，再び，新しい合計額を表示し，次に“PRINTER ON?(Y/N)”を表示して，YかNを入力すると，メニュー画面へ戻ります．

3. X，Yの項目ごとの小計

メニュー画面で2を選択すると，“X－SUM OR Y－SUM?”で，よこの小計か，たての小計かを聞いてきます．XまたはYを入力して下さい．

Xを入力すると，1からXの設定までのたての小計を出力し，最後に総計を出力して，“PRINTER ON?(Y/N)”でYまたはN ⏎ を押すと，再びメニューに戻ります．(Yで，次の小計表示のとき，プリンター出力となります)．

Yを入力した場合は，1からYの設定数までのよこの小計を出力し，同様にメニューに戻ります．

4. ソーティング

メニューで3を選択すると，たて，よこの小計の大きい順の並べかえを行ないます．まず，“SORT X OR Y?”で，Xを入力すると，たての小計の並べかえと構成比，Yを入力すると，よこの小計の並べかえと構成比を出力します．“PRINTER ON?(Y/N”でYまたは N （プリンター接続時はYを入力）を入力すると，メニューへ戻ります．

なお，このソーティング（並べかえ）中は，項目の数が多いと，出力が出るまでに時間がかかります．項目数が20で，並べかえが終るまでに1分10秒ほどかかりました．

このプログラムでは，半精度変数を多く使っていますので，多くのデータを扱えますが，入力のデータは5桁までです．

一度ソーティングして，もう一度結果をみたいときには，BRK キーを押した後，P4でRUN140 を実行して下さい．

5. 全データの出力

メニューで4を選択すると“X=1 Y=1 DATA 233”というようにデータを表示します．このとき，“PRINTER ON?(Y/N)”でYを入力してある場合には，次つぎにデータをプリントアウトしますが，そうでないときは，次のデータをみたい場合は，何かキーを押してください．次のデータを表示します．すべてのデータを表示し終わると，“PRINTER ON?(Y/N)”の表示にもどります．

プログラム

```
P0
  10 INPUT "PRINTER ON?(Y/N)",F$
  20 PRINT "1 DATA INPUT ","2 TOTAL","3
     SORT","4 DATA OUTPUT  ";
  30 INPUT R
  40 IF R=1 THEN GOTO PROG 1 ELSE IF R=
     2 THEN GOTO PROG 2
  50 IF R=3 THEN GOTO PROG 3 ELSE IF R=
     4 THEN GOTO PROG 5
  60 GOTO 10
P1
  10 INPUT "CLEAR (Y/N)?",S$
  20 IF S$="Y" THEN CLEAR :GOTO 30 ELSE
      IF S$<>"N" THEN 10 ELSE 110
  30 INPUT "X";X,"Y";Y
  40 DIM D!(X,Y),X!(X),Y!(Y)
  50 FOR J=1 TO Y:Y!(J)=0:FOR I=1 TO X
  60 PRINT "INPUT DATA X=";I;" Y=";J,
  70 INPUT D!(I,J)
  80 Y!(J)=Y!(J)+D!(I,J)
  90 NEXT I:NEXT J
 100 GOSUB 200:GOTO PROG 0
 110 PRINT "TEISEI(X,Y)";
 120 INPUT "X=";I,"Y=";J:PRINT D!(I,J)
 130 INPUT "DATA";D!(I,J)
 140 FOR J=1 TO Y:Y!(J)=0:FOR I=1 TO X
 150 Y!(J)=Y!(J)+D!(I,J)
 160 NEXT I:NEXT J
 170 GOSUB 200:GOTO PROG 0
 200 S=0:FOR I=1 TO X:X!(I)=0:FOR J=1 T
     O Y
 210 X!(I)=X!(I)+D!(I,J):S=S+D!(I,J)
 220 NEXT J:NEXT I
 230 PRINT "GRAND T.";S:FOR K=0 TO 100:
     NEXT K
 240 RETURN
```

P2

```
 10 INPUT "X-SUM OR Y-SUM";P$
 20 IF P$="Y" THEN 150 ELSE IF P$="X"
    THEN 80 ELSE 10
 80 FOR K=1 TO X
 90 PRINT "X=";K;" SUM=";X!(K)
100 IF F$="Y" THEN GOSUB 300:NEXT K:GO
    TO 340
110 FOR T=1 TO 100:NEXT T:NEXT K
120 GOTO 190
150 FOR K=1 TO Y
160 PRINT "Y=";K;" SUM=";Y!(K)
170 IF F$="Y" THEN GOSUB 320:NEXT K:GO
    TO 340
180 FOR T=1 TO 100:NEXT T:NEXT K
190 PRINT "GRAND T.";S
200 FOR K=1 TO 100:NEXT K
210 GOTO PROG 0
300 LPRINT "X=";K;" SUM=";X!(K):RETURN

320 LPRINT "Y=";K;" SUM=";Y!(K):RETURN

340 PRINT "GRAND T.";S
350 LPRINT "GRAND T.";S:GOTO PROG 0
```

P3

```
 10 PRINT "SORT X OR Y?"
 20 INPUT P$
 30 IF P$="Y" THEN GOSUB 100 ELSE IF P
    $="X" THEN GOSUB 200 ELSE 10
 40 GOTO PROG 4
100 ERASE A!
110 DIM A!(Y,2)
120 FOR J=1 TO Y
130 A!(J,1)=Y!(J):A!(J,2)=J
140 NEXT J
150 N=Y:RETURN
200 ERASE A!
210 DIM A!(X,2)
230 FOR I=1 TO X
240 A!(I,1)=X!(I):A!(I,2)=I:NEXT I
250 N=X:RETURN
```

```
P4
  10 CLS
  20 PRINT "SORTING NOW"
  30 REM SORT
  40 FOR K=N-1 TO 1 STEP -1
  50 FOR L=1 TO K
  60 IF A!(L,1)>A!(L+1,1) THEN 100
  65 FOR M=1 TO 2
  70 T=A!(L,M)
  80 A!(L,M)=A!(L+1,M)
  90 A!(L+1,M)=T
  95 NEXT M
 100 NEXT L
 110 NEXT K
 120 REM PRINT
 130 FOR K=1 TO 10:BEEP :NEXT K:CLS
 140 FOR K=1 TO N:GOSUB 220
 150 PRINT USING"##";K;" ";P$;"=";USING
     "##";A!(K,2);USING"########";A!(K,
     1);
 160 PRINT USING"###";A;CHR$(37)
 170 IF F$="Y" THEN GOSUB 300:NEXT K:GO
     SUB 320:GOTO PROG 0
 180 FOR L=1 TO 50:NEXT L:NEXT K
 190 PRINT "GRAND T.";S
 200 FOR K=0 TO 100:NEXT K
 210 GOTO PROG 0
 220 REM KOUSEIHI
 230 A=ROUND(A!(K,1)/S,-3)*100
 240 RETURN
 300 LPRINT USING"##";K;" ";P$;"=";USIN
     G"##";A!(K,2);USING"########";A!(K
     ,1);
 310 LPRINT USING"###";A;CHR$(37):RETUR
     N
 320 PRINT "GRAND T.";S
 330 LPRINT "GRAND T.";S
 340 RETURN
```

P5

```
  5 CLS
 10 FOR J=1 TO Y:FOR I=1 TO X
 20 PRINT "X=";I;" Y=";J;" DATA";D!(I,
    J)
 30 IF F$="Y" THEN GOTO 50
 40 IF INKEY$="" THEN 40 ELSE 60
 50 LPRINT "X=";I;"  Y=";J;"  DATA=";D
    !(I,J)
 60 NEXT I:NEXT J
 70 LPRINT :LPRINT :LPRINT
 80 GOTO PROG 0
```

メ　モ　リ　ー　内　容					
A	構成比	K～M	カウンタ	S＄	YまたはN
A!(　)	ソート時のデータ用	N	ソート時のデータ数	T	データ交換用変数
D!(　)	データ配列	P＄	XまたはY	X!(　)	Xの和の配列
F＄	プリンタ使用の有無	R	メニュー選択	Y!(　)	Yの和の配列
I，J	配列内添字	S	総和		

5桁以上のデータを取り扱いたいときは，プログラムのA!(　)，D!(　)，X!(　)，Y!(　)を，それぞれA(　)，D(　)，X(　)，Y(　)に変更します．

サンプルデータ

```
X= 1  Y= 1  DATA= 321
X= 2  Y= 1  DATA= 369
X= 3  Y= 1  DATA= 357
X= 1  Y= 2  DATA= 159
X= 2  Y= 2  DATA= 147
X= 3  Y= 2  DATA= 123
X= 1  Y= 3  DATA= 842
X= 2  Y= 3  DATA= 862
X= 3  Y= 3  DATA= 579
```

操 作

	手順	キー操作	表　示
プリンターを使いますか？→		[SHIFT] [PO/Ø]	PRINTER ON ? (Y/N)
	1	N ↵	1 DATA INPUT 2 TOTAL 3 SORT 4 DATA OUTPUT ?_
[データ入力] データをクリアしますか？→	2	1 ↵	CLEAR(Y/N)
よこの項目数は？→	3	Y ↵	X ?
たての項目数は？→	4	3 ↵	Y ?
X=1, Y=1のデータを入力して下さい→	5	3 ↵	INPUT DATA X= 1 Y= 1 ?
X=2, Y=1のデータを入力して下さい→	6	321 ↵ ⋮	INPUT DATA X= 2 Y= 1 ?
		以下続けてデータを入力しています.	
総計の表示 プリンターを接続していればY, 否ならばN→	7		GRAND T. 3759 PRINTER ON ? (Y/N)
メニューに戻り, 命令まち.→	8	N ↵	1 DATA INPUT 2 TOTAL 3 SORT 4 DATA OUTPUT ?_
[たての小計] たて小計ならばX, よこ小計ならばY→	9	2 ↵	X−SUM OR Y−SUM ?
各小計と合計の出力.→ SUMは合計です. プリンター接続ならばY→	10	X ↵	X=1 SUM=1322 X=2 SUM=1378 X=3 SUM=1059 GRAND T. 3759 PRINTER ON ? (Y/N)
メニューに戻る	11	N ↵	1 DATA INPUT 2 TOTAL 3 SORT 4 DATA OUTPUT ?_
[よこの小計]	12	2 ↵	X−SUM OR Y−SUM ?

	手順	キー操作	表　示
よこの小計はY ⟶	13	Y ⏎	Y=1 SUM=1047 Y=2 SUM=429 Y=3 SUM=2283 GRAND T. 3759
プリンター接続ならばY ⟶	14		PRINTER ON ? (Y/N)
	15	N ⏎	1 DATA INPUT 2 TOTAL 3 SORT 4 DATA OUTPUT
ソーティング たて(X), よこ(Y)の小計ごとの並べかえ	16	3 ⏎	SORT X OR Y ?
たて小計の大きい順に表示. ⟶ 順位, 項目名, 小計, %を表示.	17	X ⏎	1 X=2 1378 37% 2 X=1 1322 35% 3 X=3 1059 28% GRAND T. 3759
プリンター接続ならば, Y ⟶	18		PRINTER ON ? (Y/N)
メニューに戻る ⟶	19	Y ⏎	1 DATA INPUT 2 TOTAL 3 SORT 4 DATA OUTPUT
	20	3 ⏎	SORT X OR Y ?
よこ小計の並びかえをしたいときはY. ⟶ 小計の大きい順に表表示し. %も出力.	21	Y ⏎	1 Y=3 2283 61% 2 Y=1 1047 28% 3 Y=2 429 11% GRAND T. 3759
	22		PRINTER ON ? (Y/N)
	23	4 ⏎	X=1 Y=1 DATA= 321 X=2 Y=1 DATA= 369 ⋮

※尚, データの数値により, 構成比の総和が100%とならないことがあります.

バイオリズム

自分の誕生日から，ある月のバイオリズムを知ることができます．身体，感情，知性の曲線が，目盛り上にあるときが一番不安定な状態で，それよりも上にいくほど状態がよく，下にいくほど悪くなります．
つづけて次の月のデータも知ることができます．

使い方

プログラムをスタートさせると，まずプロッタに出すかどうかをきいてきますから，使用する場合はY，使用しなければNを押します．

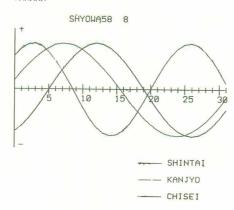
次に名前をきいてきますので，入力します(30文字まで入ります)．

今度は誕生日をきいてきますから，昭和の場合はS，大正の場合はT，明治の場合はMで入力します．西暦で入れたいときは0から9までの数を押してから，年号を入れます．同じように，月と日を入力します．

調べたい年と月をきいてきますから，同じ要領で指定します．

入力後，画面上に指定した月のバイオリズムが，身体，感情，知性の順で描かれます．プロッタプリンタにつないであれば，身体が青，感情が緑，知性が赤の色で出力されます．

ひとつの月のバイオリズムが出力されると，次の月のデータの出力をきいてきます．ここでNを押すと，プログラムの初めに戻り，新たに名前を入力していくことになります．

プリントアウト例

プログラム

```
100 REM
110 CLEAR :DIM G$(3),S$(3),K(3),C$(1),
    N$(0)*30
120 GOSUB 2000
130 REM
140 CLS :PRINT MD$:INPUT "Printer(Y/N)
    :",ZI$:PS=0:IF ZI$="Y" THEN PS=1
200 REM
210 CLS :PRINT MD$:INPUT "NAME:",N$(0)
220 REM
230 PRINT "S/T/M/(0-9)"
240 SW=0:GOSUB 1000:BG=G
250 SW=1:GOSUB 1000:OG=G:IF OY<BY THEN
     250 ELSE IF OY=BY THEN IF OM<BM T
    HEN 250
300 REM
310 NE=BY:TK=BM:GOSUB 1200:BS=NI
320 NE=OY:TK=OM:GOSUB 1200:OS=NI:TU=OS
    -BS-BD
330 NE=OY:TK=OM+1:GOSUB 1200:TN=NI-OS
400 REM
410 IF OG=0 THEN ZI$="" ELSE ZI$=G$(OG
    )
420 CN$=ZI$+MID$(STR$(OY-K(OG)),2)+" "
    +RIGHT$(STR$(OM),2)+" "
430 CLS
440 DRAW(0,0)-(0,32):DRAW(0,16)-(93,16
    )
450 FOR I=1 TO TN:Y=1:IF I MOD 5=0 THE
    N Y=2
460 DRAW(I*3,16-Y)-(I*3,16+Y):NEXT I
470 P=TU MOD 23:A=P:W=23:LOCATE 12,0:P
    RINT S$(1):GOSUB 1300
480 S=TU MOD 28:A=S:W=28:LOCATE 12,1:P
    RINT S$(2):GOSUB 1300
490 H=TU MOD 33:A=H:W=33:LOCATE 12,2:P
    RINT S$(3):GOSUB 1300
500 REM
510 IF PS=1 THEN GOSUB 1400:GOTO 540
```

```
 520 IF INKEY$="" THEN 520
 530 REM
 540 OM=OM+1:IF OM>12 THEN OM=1:OY=OY+1
 550 CLS :PRINT MD$;OY;" ";OM;"
     ":INPUT " (Y/N)",ZI$
 560 IF ZI$="Y" THEN 300 ELSE 200
1000 REM
1010 PRINT C$(SW)
1020 ZI$=INKEY$:IF ZI$="" THEN 1020
1030 IF ZI$="S" THEN G=1:GOTO 1060
1040 IF ZI$="T" THEN G=2:GOTO 1060
1050 IF ZI$="M" THEN G=3 ELSE G=0
1060 PRINT "  => ";G$(G);" ";
1070 IF SW=1 THEN INPUT "",OY,OM:IF OY<
     1 THEN 1060 ELSE IF OM<1 THEN 1060
1075 IF SW=1 THEN IF OM>12 THEN 1060 EL
     SE OY=OY+K(G):GOTO 1090
1080 INPUT "",BY,BM,BD:IF BY<1 THEN 106
     0 ELSE IF BM<1 THEN 1060 ELSE IF B
     M>12 THEN 1060
1085 IF BD<1 THEN 1060 ELSE IF BD>31 TH
     EN 1060 ELSE BY=BY+K(G)
1090 RETURN
1200 REM
1210 U=NE-1:NI=U*365+INT(U/4)-INT(U/100
     )+INT(U/400)
1220 RESTORE 1270:FOR I=1 TO TK:READ X:
     NI=NI+X:NEXT I
1230 IF NE MOD 4>0 THEN 1250 ELSE IF NE
      MOD 100=0 THEN IF NE MOD 400>0 TH
     EN 1250
1240 IF TK>2 THEN NI=NI+1
1250 RETURN
1260 REM
1270 DATA0,31,28,31,30,31,30,31,31,30,3
     1,30,31
1300 REM
1320 FOR I=0 TO TN:Y=2.5*SIN((A+I)/W*36
     0)*5:DRAW(I*3,16-Y):NEXT I
1330 RETURN
1400 REM
1410 IF BG=0 THEN ZI$="" ELSE ZI$=G$(BG
     )
```

```
1420 UM$=LEFT$(ZI$,1)+MID$(STR$(BY-K(BG
     )),2)+" "+RIGHT$(STR$(BM),2)+" UMA
     RE"
1430 LPRINT CHR$(28);CHR$(37):LPRINT "P
     ";UM$:LPRINT "H5":LPRINT "Z2,6"
1440 X=12:Y=-3.6:LPRINT "S2":LPRINT "M"
     :X;",0":LPRINT "M,";Y*2
1450 LPRINT "S1":LPRINT "M,";Y*2:LPRINT
      "P";N$(0)
1460 LPRINT "M";X*2;",";Y*6:LPRINT "P";
     CN$
1470 LPRINT "O0,-50":LPRINT "D0,25,0,-2
     5":FOR I=1 TO TN:X=I*3:Y=1:IF I MO
     D 5=0 THEN Y=2
1480 LPRINT "D";X;",";Y;",";X;",";-Y:NE
     XT I:LPRINT "D";X;",0,0,0"
1490 LPRINT "M2,24":LPRINT "P+":LPRINT
     "M2,-25":LPRINT "P-"
1500 FOR I=5 TO TN STEP 5:X=(I-1)*3:LPR
     INT "M";X;",-5":LPRINT "P";I:NEXT
     I
1510 J=1:A=P:W=23:GOSUB 1600:J=2:A=S:W=
     28:GOSUB 1600:J=3:A=H:W=33:GOSUB 1
     600
1520 FOR I=1 TO 3:Y=-(I*7.2+25):LPRINT
     "M55,";Y+0.8:LPRINT "J";I:LPRINT "
     D,65,";Y+0.8
1530 LPRINT "M68,";Y:LPRINT "P";S$(I):N
     EXT I
1540 LPRINT CHR$(28);CHR$(46);CHR$(27);
     "J0"
1550 RETURN
1600 REM
1610 Y=SIN(A/W*360)*20:LPRINT "M0,";Y:L
     PRINT "J";J
1620 FOR I=1 TO TN:X=I*3:Y=SIN((A+I)/W*
     360)*20:LPRINT "D,";X;",";Y:NEXT I
1630 RETURN
2000 REM
2010 RESTORE 2100:FOR I=0 TO 3:READ G$(
     I),S$(I),K(I):NEXT I
2020 C$(0)="TANJYOBI "
2030 C$(1)="SHIRABETAI TSUKI"
2040 MD$=" ** BIORHYTHM **"
```

```
2050 RETURN
2100 REM
2110 DATA SEIREKI,,0
2120 DATASHYOWA,SHINTAI,1925
2130 DATATAISHYO,KANJYO,1911
2140 DATAMEIJI,CHISEI,1867
```

操作

	手順	キー操作	表示
プロッタに出力するかどうか →			＊＊BIORHYTHM＊＊ Printer(Y/N):
	1	Y or N	NAME:
名前を入力 →	2	TANAKA ⏎	NAME:TANAKA S/T/M/(0−9) TANJYOBI
誕生日を入力 →	3	Ⓢ 32 ⏎ 5 ⏎ 17 ⏎	=>SHYOWA 32 ?5 ?17 SHIRABETAI TSUKI
調べたい年月を入力 →	4	Ⓢ 58 ⏎ 8 ⏎	=>SHYOWA 58 ?8
プロッタに出力 → プロッタに出力.次の月を調べるかどうかをきいてくる →	5		＊＊BIORHYTHM＊＊ 1983 9 (Y/N)
	6	Y or N	

メモリー内容			
A	月の最初のバイオリズムの状態	MI	日数
BD BM BY	誕生年月日	OM OY	調べる月,年
G$( )	年号	PS	プリンタ使用有無
MD$	"＊＊BIORHYTHM＊＊"	W	バイオリズムの周期
NA$	名前	X Y	グラフのX,Y座標
NE	年数	ZI$	一文字入力および年号

汎用グラフソフト

プロッタプリンタを用いてグラフを書くプログラムです．データは最大12個まで，またデータの値は，次の範囲に限定されます．

|データの値|≦1E90

4色のカラープロッタの特長を生かし，棒，帯，折れ線グラフをきれいに描きます．

使い方

プログラムをスタートさせると，まずメニューが表示されます．最初はデータの入力をしますので，[1]を入力します．

データは，上に示した範囲の数で，負の数でもかまいません．また，12個目のデータを入力すると，自動的にメニュー表示に戻ります．途中で入力をやめるときは ↵ キーを押します．

[2]は，データの修正です．入力したデータが先頭から表示されますので，修正する個所で正しい数を入力して下さい． ↵ キーを押すと次のデータになり，[SHIFT] キーと ↵ キーを同時に押すと前のデータに戻ります．

[3]はグラフを書くルーチンです．[3]を押しますと，1から4までのグラフの種類が表示されます(ただし，4は最初のメニューに戻ります)．

1は，帯グラフで，これは全体を100％として，それぞれのデータの割合を示します．見やすいように，となりのデータとは違った色で出力されます．またこのグラフは，データに負の数があった場合，出力されません．

2は棒グラフです．スケールは，入力されたデータに応じてコンピュータが設定してくれます．正の値は緑，負の値は赤で出力されます．

3は，折れ線グラフです．

また3を入力すると，“Over Previous Graph ? ”ときいてきますが，Yを押せば，すぐ前に棒グラフが書いてある場合は，それに重ねてグラフを書きます．また別にグラフを書きたい場合や，すぐ前に棒グラフが書いていない場合は，Nを入力します．スケールを作り，単独に折れ線グラフを書きます．

どのグラフも，書き終えると，最初のメニューに戻ります．それからプログラムをスタートさせると，データがクリアされるので，注意して下さい．

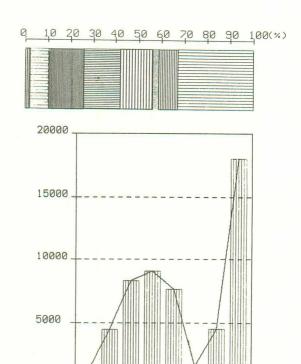
[4]は，ENDで，プログラムの実行が終わります．

[5]は，入力したデータを，プロッタに出力します．また合計も出ます．

プリンアウト例

```
Print Data

D(1)= 1200
D(2)= 4500
D(3)= 8383
D(4)= 9102
D(5)= 7701
D(6)= 1532
D(7)= 4562
D(8)= 18020

Total= 55000
```

プログラム

```
 10 CLEAR
 20 CLS :PRINT "    ---- DATA ----"
 30 PRINT TAB(2);"1:Input 2:Correct",T
    AB(2);"3:Graph 4:END",TAB(2);"5:Pr
    Int Data";
 40 K=VAL(INKEY$):IF K<1 THEN 40 ELSE
    IF K>5 THEN 40
 50 BEEP :GOTO K*100
100 CLS :ERASE D:DIM D(13):Z=1
110 LOCATE 2,3:PRINT "<RETURN> : END";
120 LOCATE 6,0:PRINT "              ";
130 LOCATE 0,0:PRINT "D(";Z;")=";:INPU
```

```
    T "",AB$
140 IF AB$<>"" THEN D(Z)=VAL(AB$):IF Z
    <12 THEN Z=Z+1:GOTO 120 ELSE 20 EL
    SE Z=Z-1:GOTO 20
200 IF Z<1 THEN 20
210 CLS :I=1:PRINT TAB(46);":Shift RET
    URN",TAB(6);":RETURN ";
220 LOCATE 4,2:PRINT CHR$(228);CHR$(22
    9):LOCATE 4,3:PRINT CHR$(230);CHR$
    (231);
230 DRAWC(32,23)-(47,23)-(47,24)-(32,2
    4):LOCATE 0,0:PRINT "D(";
240 LOCATE 2,0:PRINT I;")=";D(I);"
              ":LOCATE 6,1:PRINT "
              "
250 LOCATE 6,1:INPUT AB$:K$=INKEY$:IF
    AB$<>"" THEN D(I)=VAL(AB$)
260 IF K$=CHR$(24) THEN I=I-1:IF I<1 T
    HEN 20 ELSE 240
270 IF K$=CHR$(13) THEN I=I+1:IF I>Z T
    HEN 20 ELSE 240
280 IF K$=CHR$(23) THEN I=I+1:IF I>Z T
    HEN 20 ELSE 240
290 IF K$="" THEN I=I+1:IF I>Z THEN 20
     ELSE 240
300 IF Z<1 THEN 20 ELSE CLS
310 PRINT TAB(5);"1:Band",TAB(5);"2:Ba
    r"
320 PRINT TAB(5);"3:Line",TAB(5);"4:ME
    NU";
330 K=VAL(INKEY$):IF K<1 THEN 330 ELSE
     IF K>4 THEN 330
340 BEEP :GOTO K*1000
400 LPRINT CHR$(28);CHR$(46):END
500 IF Z<1 THEN 20 ELSE GOSUB 6000:LPR
    INT "Q1":V=0
510 LPRINT "M93,0","PPrint Data"
520 FOR I=1 TO Z
530 LPRINT "M";90-4*I;",0","PD(";MID$(
    STR$(I),2);")=";D(I)
540 V=V+D(I):NEXT I
550 LPRINT "M";90-I*4-5;",0","PTotal="
    ;V:GOSUB 6000:GOTO 20
```

```
1000 GOSUB 6000:A=0
1010 FOR I=1 TO Z:IF D(I)<0 THEN ERASE
     I:GOTO 20 ELSE A=A+D(I):NEXT I
1020 IF A<=0 THEN 20
1030 LPRINT "Q5,0","X1,8,10","M85,1","P
     (";CHR$(37);")"
1040 FOR I=100 TO 0 STEP -10:LPRINT "M"
     ;-4+8*I/10;",1","P";I:NEXT I
1050 B=0:C=0
1060 FOR I=1 TO Z
1070 C=ROUND(D(I)/A*80+C,-2)
1080 LPRINT "J";I MOD 4,"A";B;",-3,";C;
     ",-23"
1090 LPRINT "G";I MOD 2+1;",";C-B;",-20
     ,";(I MOD 3)/4+.5
1100 B=C:NEXT I
1110 LPRINT "H30":GOTO 20
2000 GOSUB 6000:GOSUB 7000:GOSUB 8000
2010 FOR I=1 TO Z
2020 IF D(I)>=0 THEN J=2 ELSE J=3
2030 LPRINT "J";J
2040 LPRINT "A";O;",";6-8*I;",";ROUND(D
     (I)/A*90/N+O,-2);",";-8*I
2050 LPRINT "M";O;",";6-8*I,"G1,";ROUND
     (D(I)/A*90/N,-2);",-6"
2060 NEXT I
2070 LPRINT "M0,";-8*(Z+1):GOTO 20
3000 IF Z<2 THEN 20 ELSE CLS :PRINT "Ov
     er previous graph?",TAB(8);"Y/N"
3010 K$=INKEY$:IF K$="Y" THEN BEEP :GOT
     O 3080 ELSE IF K$<>"N" THEN 3010 E
     LSE BEEP
3020 GOSUB 6000
3030 GOSUB 7000:GOSUB 8000
3040 LPRINT "L1","J0"
3050 FOR I=1 TO Z
3060 F=3-8*I:IF I MOD 2=1 THEN LPRINT "
     D0,";F;",90,";F ELSE LPRINT "D90,"
     ;F;",0,";F
3070 NEXT I:LPRINT "L0","M0,0"
3080 S=S+1:IF S>3 THEN S=0
3090 T=T+.25:IF T>3.9 THEN T=0
```

```
3100 IF T<2 THEN LPRINT "B1.6" ELSE LPR
     INT "B6.4"
3110 LPRINT "J";S,"L";T
3120 G=ROUND(D(1)/A*90/N+O,-2):H=-5
3130 FOR I=2 TO Z
3140 U=ROUND(D(I)/A*90/N+O,-2):V=3-8*I
3150 LPRINT "D";G;",";H;",";U;",";V
3160 G=U:H=V
3170 NEXT I
3180 LPRINT "B3.2","M0,";-8*(Z+1):GOTO
     20
4000 GOTO 20
6000 LPRINT CHR$(28);CHR$(37),"O0,0","J
     0","L0","S1","Q0","Y0","B3.2","H20
     "
6010 S=0:T=0:RETURN
7000 Y=-9E99:B=9E99
7010 FOR I=1 TO Z
7020 IF D(I)>Y THEN Y=D(I)
7030 IF D(I)<B THEN B=D(I)
7040 NEXT I
7050 IF Y>=0 THEN D(0)=Y ELSE D(0)=0
7060 IF B>0 THEN D(Z+1)=0 ELSE D(Z+1)=B
7070 RETURN
8000 IF SGND(0)*SGND(Z+1)<=0 THEN M=ABS
     (D(0)-D(Z+1)) ELSE M=D(0):IF M<0 T
     HEN M=ABSD(Z+1)
8010 IF M=<0 THEN 20 ELSE R=INTLGTM:A=1
     0^R
8020 IF A*INT(M/A)*.75<M THEN A=A*.5
8030 D=LEN(STR$(A))*2.4+5
8040 IF SGND(0)*SGND(Z+1)<0 THEN N=INT(
     M/A)+2 ELSE N=INT(M/A)+1
8050 C=ABSINT(D(0)/A)+SGND(0)
8060 FOR I=N TO 0 STEP -1:IF C=0 THEN O
     =I*90/N
8070 C=C-1:NEXT I
8080 LPRINT "D0,0,90,0","Q1":W=18/N:V=0
8090 V=V+5:IF V*W<18 THEN 8090
8100 IF D(0)<=0 THEN 8150 ELSE X=O
```

```
8110 K=ROUND(X,-2):LPRINT "D";K;",2,";K
     ;",-2"
8120 LPRINT "M";K;",";D,"P";ROUND((X-O)
     *A*N/90,R-2)
8130 LPRINT "L1","D";K;",0,";K;",";-Z*8
     -2,"L0":X=X+W*V
8140 IF X<90 THEN 8110
8150 LPRINT "D90,2,90,-2","M90,";D,"P";
     ROUND((90-O)*A*N/90,R-2)
8160 IF D(Z+1)>=0 THEN 8220 ELSE X=O-W*
     V
8170 K=ROUND(X,-2):LPRINT "D";K;",2,";K
     ;",-2"
8180 LPRINT "M";K;",";D,"P";ROUND((X-O)
     *A*N/90,R-2)
8190 LPRINT "L1","D";K;",0,";K;",";-Z*8
     -2,"L0":X=X-W*V
8200 IF X>0 THEN 8170
8210 LPRINT "D0,2,0,-2","M0,";D,"P";ROU
     ND(-O*A*N/90,R-2)
8220 LPRINT "D0,0,0,";-8*Z-2;",90,";-8*
     Z-2;",90,0","M0,0"
8230 RETURN
```

操 作

	手順	キー操作	表　示
		[SHIFT] [P0 Ø]	---- DATA ---- 1：Input　2：Correct 3：Graph　4：End 5：Print Data
[1]データの入力	1	[1]	D(　1)= 〈RETURN〉：END
	2	2112 ↵ ⋮	D(　2)= ⋮
入力の中断→ (12まで入力するとメニューへ戻る)	3	↵	
[2]データの訂正 データが終わるとメニュー→	4	[2] 1200 ↵	D(　1)=1200 ? ▲Shift RETURN (前のデータ) ▼RETURN (次のデータ)
[3]グラフ作成	5	[3]	1：Band　(帯グラフ) 2：Bar　(棒グラフ) 3：Line　(折れ線グラフ) 4：MENU　(メニュー)
帯グラフがプロッタに出力→ 棒グラフがプロッタに出力→ 前の棒グラフに折れ線グラフを重ねるか→ 折れ線グラフをプロッタに出力→	6	………[1] ………[2] ………[3] Y or N	(出力後最初のメニューへ) (出力後最初のメニューへ) Over Previous Graph ? Y／N (出力後最初のメニューへ)
[5]データ出力 データをプロッタに出力→	7	[5]	(出力後最初のメニューへ)
[4]実行を終了	8	[4]	

巻末資料編

PB−700命令一覧表

演算記号

1 算術演算子

名　称	一般の使い方	使　い　方	意　　　味	計算の優先順位
べき算	x^y	X∧Y	XのY乗	1
かけ算	$x \times y$	X＊Y	XとYを掛ける	2
わり算	$x \div y$	X／Y	XをYで割る	2
剰　余	$x \div y = z \cdots$	X MOD Y	X÷Yの余り	3
たし算	$x + y$	X＋Y	XとYを足す	4
ひき算	$x - y$	X−Y	XからYを引く	4
代　入	$x = y + 5$	X＝Y＋5	XにY＋5を代入	5

2 関係演算子（条件式）

一般の使い方	使　い　方	意　　　味
$x = y$	X＝Y	XとYが等しい
$x \neq y$	X<>Y, X><Y	XとYが等しくない
$x < y$	X<Y	XはYより小さい
$x > y$	X>Y	XはYより大きい
$x \leqq y$	X<＝Y, X＝<Y	XはYより小さいか等しい
$x \geqq y$	X>＝Y, X＝>Y	XはYより大きいか等しい

- 関係演算子はIF文においてのみ有効
- 数値どうしは定数，変数，式が，文字どうしは定数と変数が比較できる．

3 文字式演算子

＋……文字列は ＋ により連結することができる．

特殊文字

一般の使い方	使　い　方	意　　　味
$x \times 10^y$	X E Y	指数部置数，10のY乗

- 演算結果が10^{10}以上または10^{-3}未満の場合，自動的に指数表示となる．

数値関数

名　称	一般の使い方	コマンド使用例	意　味，注　意
三角関数	sin x cos x tan x	SIN X COS X TAN X	Xの正弦sinXを与える(−5400<X<5400) Xの余弦cosXを与える(−5400<X<5400) Xの正接tanXを与える(−5400<X<5400)
逆三角関数	$\sin^{-1}$ x $\cos^{-1}$ x $\tan^{-1}$ x	ASN X ACS X ATN X	逆正弦$\sin^{-1}$Xを与える（|X|≦1） 逆余弦$\cos^{-1}$Xを与える（|X|≦1） 逆正接$\tan^{-1}$Xを与える（|X|<10^{100}）
対数関数	log x ln x	LGT X LOG X	$\log_{10}$X（常用対数）(X>0) $\log_{e}$X（自然対数）(X>0)
指数関数	e^x x^y	EXP X X∧Y	e^x（X≦230） XのY乗計算
平方根	$\sqrt{x}$	SQR X	平方根を与える（X≧0）
絶対値化	$\|x\|$	ABS X	Xの絶対値を得る
整数化		INT X	Xを越えない最大の整数を与える。 〔例〕 INT 1.2 → 1 INT−1.2 → −2
整数部除去		FRAC X	小数点以下の数値のみ取り出す。絶対値表示ではないので±の符号はつく。
円周率	π	PI	円周率の概数を得る。与えられるPIの値は11桁で π=3.1415926536
乱数		RND	0<RND<1の擬似乱数を発生。与えられる乱数は10桁。
符号化		SGN X	引数の符号を調べる X<0なら−1 X=0なら　0 X>0なら　1 を得る
四捨五入		ROUND(X,Y)	Xを10のY乗の位で四捨五入する。Y<100

文字関数

用　途	コマンド	使用例	使用例の意味
先頭1文字の10進コードを得る	ASC	PRINT ASC ("E")	キャラクターEの10進コードを表示する.
キャラクターコードの1文字を得る	CHR$	PRINT CHR$ (69)	キャラクターコード69の文字Eを表示する.
文字列の数字を数値に変換	VAL	A＝VAL(X$)	文字変数X$に記憶されている数字の文字列を数値に変換.
数値を文字列に変換	STR$	C$＝STR$(X)	数値変数Xに記憶されている数値を数字の文字列に変換.
文字列の左から指定文字数分を取り出す	LEFT$	C$＝LEFT$ (X$,3)	X$に記憶されている文字列の左から3文字を取り出しC$に代入.
文字列の右から指定文字数分を取り出す	RIGHT$	C$＝RIGHT$ (X$,3)	X$に記憶されている文字列の右から3文字を取り出しC$に代入.
文字を指定した数だけ取り出す	MID $	C$＝MID$(X$, 3;5)	X$に記憶されている文字列の左から3番目より5文字を取り出しC$に代入.
文字列の文字数を求める	LEN	A＝LEN(X$)	X$に記憶されている文字列の文字数をAに代入.
キーボードからの1文字入力	INKEY$	A$＝INKEY$	INNEY$を実行したとき押されているキーがあればA$へ代入する. 代入できるのは1文字.

表示用関数

用　途	コマンド	使用例	使用例の意味
画面上のドットが点燈しているか否かを与える	POINT	POINT(10,20)	画面の座標(10, 20)にドットが点燈しているかどうかをみる.点燈は1, 点燈していない場合は0を与える.
指定された桁数だけカーソルを移動する	TAB	PRINT TAB(10)	画面の右方向10桁目までスペースを出力する.
表示形式を指定する	USING	PRINT USING "＃＃＃.＃＃" ;A	数値変数Aに記憶されている数値を〝＃＃＃.＃＃″のフォーマットで表示する.

マニュアル・コマンド

用　途	コマンド	使用例	使用例の意味
プログラムの実行再開	CONT	CONT	STOP文もしくは STOP/ANS キーによって停止したプログラムの実行を再開する.
プログラムの削除	DELETE	DELETE 50	50行を削除する.
		DELETE 30−	30行以後を削除する.
		DELETE −100	100行以前を削除する.
		DELETE 150−200	150行から200行までを削除する.
プログラムの修正	EDIT	EDIT	最初の行を表示してEDITモードにする.
		EDIT 30	30行を表示してEDITモードにする.
プログラムのリスト	LIST	LIST	現在指定されているプログラムエリアのプログラムを先頭行から表示する.
		LIST 50	50行を表示する.
		LIST 30−	30行以後を表示する.
		LIST −100	100行以前を表示する.
		LIST 150−200	150行から200行までを表示する.
		LIST ALL	全エリアのプログラムを表示する.
		LIST V	登録済変数名を表示する.
プログラムの印字	LLIST	LLIST	現在指定されているプログラムエリアのプログラムを印字する.
		LLIST 50	50行を印字する.
		LLIST 30−	30行以後を印字する.
		LLIST −100	100行以前を印字する.
		LLIST 150−200	150行から200行までを印字する.
		LLIST ALL	全プログラムエリアのプログラムを印字する.
		LLIST V	登録済変数名を印字する.
カセットからプログラムを読み込む	LOAD	LOAD	現在指定されているプログラムエリアに，内部コード方式のプログラムを読み込む.
		LOAD ALL	全プログラムエリアに内部コード方式のプログラムを読み込む.

用　途	コマンド	使　用　例	使　用　例　の　意　味
カセットからプログラムを読み込む	**LOAD**	LOAD, A	現在指定されているプログラムエリアに，アスキーコード形式のプログラムを読み込む．
		LOAD, M	現在指定されているプログラムエリアのプログラムとアスキーコード形式のプログラムを混合する．
		LOAD "ABC" LOAD ALL "CASIO" LOAD "TEST", A LOAD "TEXT", M	指定ファイル名のプログラムに対して上記書式と同様の動作を行なう．
プログラムの消去	**NEW**	NEW	現在指定されているプログラムエリアのプログラムを消去する．
		NEW ALL	全**RAM**を消去して初期状態に設定する．
プログラムの保護	**PASS**	PASS "KEY"	"**KEY**" という名でパスワードを設定する．
プログラムエリアの指定	**PROG**	PROG 2	プログラムエリア**P** 2を指定する．
プログラムの実行開始	**RUN**	RUN	現在指定されているプログラムエリアの先頭から実行を開始する．
		RUN 100	100行目からプログラムを実行開始する．
カセットにプログラムを書き込む	**SAVE**	SAVE	現在指定されているプログラムエリアのプログラムを内部コード形式で書き込む．
		SAVE ALL	全プログラムのプログラムを内部コード形式で書き込む．
		SAVE, A	現在指定されているプログラムエリアのプログラムをアスキーコード形式で書き込む．
		SAVE "ABC" SAVE ALL "CASIO" SAVE "TEST", A	プログラムにファイル名をつけて上記書式と同様の動作を行う．
プログラムの使用状態表示	**SYSTEM**	SYSTEM	プログラムエリアの使用状態，残り**Byte**数，設定**ANGLE**を表示する．

用　途	コマンド	使用例	使用例の意味
カセットのプログラムのチェック	**VERIFY**	VERIFY	最初に出現したプログラムファイルのパリティチェックを行う.
		VERIFY "ABC"	指定ファイル名のプログラムに対して上記書式と同様の動作を行う.

プログラム・コマンド

※印のついたコマンドはマニュアルでも使えます.

用　途	コマンド	使用例	使用例の意味
角度単位指定※	**ANGLE**	ANGLE 0	角度単位を度(テグリー)に指定.
		ANGLE 1	角度単位をラジアンに指定.
		ANGLE 2	角度単位をグラジアンに指定
ブザー音を出力※	**BEEP**	BEEP	**BEEP** 0と同じ.
		BEEP 0	ブザー音を発生(ブ), 1回.
		BEEP 1	ブザー音を発生(ピ), 1回.
プログラムを読み込み実行※	**CHAIN**	CHAIN	最初に出現した**PF B**を**LOAD**し実行する.
		CHAIN "XYZ"	指定ファイル名のプログラムに対し上記と同様に実行する.
変数の消去※	**CLEAR**	CLEAR	全ての変数をクリアする.
表示の消去	**CLS**	CLS	表示を全てクリアし, ホーム位置にカーソルを移す.
データの格納※	**DATA**	DATA 1,2,3	**READ**文で参照するデータを格納する.
配列の宣言※	**DIM**	DIM A(3)	1次元単精度数値配列を宣言する.
		DIM B(2,3)	2次元単精度数値配列を宣言する.
		DIM C!(4)	1次元半精度数値配列を宣言する.
		DIM D!(3,4)	2次元半精度数値配列を宣言する.
		DIM G$(2)＊3	1次元文字配列を宣言し文字長3を指定する.
		DIM H$(4,5)＊6	2次元文字配列を宣言し文字長3を指定する.
点と直線を描く	**DRAW**	DRAW (0,0)	座標(0, 0)に点を描く.
		DRAW (1,0)－(5,10)	座標(1, 0)から(5, 10)に線を描く.
点と直線を消す	**DRAWC**	DRAWC (0,0)	座標(0, 0)の点を消す.
		DRAWC (1,0)－(5,10)	座標(1, 0)から(5, 10)の線を消す.

用　途	コマンド	使用例	使用例の意味
プログラムの実行終了	END	END	プログラムの実行を終了する.
配列名の削除*	ERASE	ERASE A	Aという登録配列変数を削除する.
ループ（繰り返し）	FOR TO STEP ⌇ NEXT	FOR I=5 TO 20 STEP 0.5 ⌇ NEXT I	FORとNEXTに囲まれた処理を，変数Iが5から始まり1回に0.5ずつ足して20を越えない範囲で繰り返し実行する.
カセットから変数データを読み込む*	GET	GET A	最初に出現した変数データを読み込む.
		GET "MAX" B	"MAX"というファイル名の変数データを読み込む.
サブルーチンへ分岐	GOSUB	GOSUB 100	100行のサブルーチンへ分岐.
		GOSUB PROG 3	プログラムエリアP 3のサブルーチンへ分岐.
サブルーチンの終わり	RETURN	RETURN	GOSUB文の次の命令に復帰する.
無条件分岐	GOTO	GOTO 500	500行に分岐.
		GOTO PROG 5	プログラムエリアP 5へ分岐.
条件分岐	IF条件式 THEN～ ELSE～	IF I>9 THEN 50 ELSE 80	Iが9より大きいなら50行へ分岐それ以外は80へ分岐.
データをキーから入力	INPUT	INPUT S	？を表示後Sに入力を待つ.
		INPUT "NAME", T$	NAME 表示後T$に入力を待つ.
		INPUT "NAME"; U$	NAME？表示後U$に入力を待つ.
データを変数に代入	LET	LET A=B	AにBを代入する.
カーソルの位置指定	LOCATE	LOCATE 2,3	座標(2，3)にカーソルを指定.
データを表示*	PRINT	PRINT C,D	CとDの値を1行ごとに表示.
		PRINT C;D	CとDの値をつづけて表示.
データを印字*	LPRINT	LPRINT C,D	CとDの値を1行ごとに印字.
		LPRINT C;D	CとDの値をつづけて印字.
カセットに変数データを書き込む*	PUT	PUT A	変数Aのデータを書き込む.
		PUT "DATA" A	ファイル名をつけて変数Aのデータを書き込む.

用　途	コマンド	使用例	使用例の意味
格納されたデータを読み込む	**READ**	READ X	変数XにDATA文によって格納されたデータを読み込む.
注釈文	**REM**	REM * * *	プログラムに注釈を書き込む.
DATA文の実行順変更	**RESTORE**	RESTORE	次のREAD文で最初のDATA文から読み始める.
		RESTORE 100	次のREAD文で100行目のDATA文から読み始める.
実行の中断	**STOP**	STOP	実行を中断する.
実行の追跡*	**TRON**	TRON	プログラムの実行状態を追跡する.
実行の追跡の解除*	**TROFF**	TROFF	トレースモードを解除する.

エラーメッセージ一覧表

エラーメッセージ	エラーの内容	対　策
BS error (Bad Subscript)	① 配列変数の添字の値が負，または256以上ある. 例）**DIM A(256)** ② 指定数値が引数範囲を外れている. 例）**POINT**コマンドの範囲は0≦X≦159, 0≦Y≦31だが,この範囲を外れている.	① 添字の値を規定範囲内にする. 添字が変数の場合は関連の個所を調べる. ② 引数範囲内に指定変更する.
BV error (Buffer oVerflow)	① 入出力バッファがオーバーフローした.	① 演算，プログラムの1文とも79文字以内に収める.
DA error (read without DAta)	① 読むべきデータがないのに**READ**文及び**GET**が実行された.	① **READ**文と**DATA**文の関係を確認する. 変数に代入すべきデータを文中に作る.
DD error (Duplicate Definition)	① 同一配列名で添字の異なるものを二重に定義した. 例）**DIM**文の配列変数宣言でエラーとなるのは…… **DIM A(1), A(2,3)** **DIM A!(1), A!(2,3)** **DIM A$(1), A$(2,3)** **DIM A$(1)*20, A$(2,3)*20** **DIM A$(1)*20, A$(2,3)*20**	① エラーが発生した行の変数を調べ、同一配列名の添字を点検する. 添字の異なっているものがあればどちらか一方の配列名を変更し，プログラムの再構成を行なう.

エラーメッセージ	エ ラ ー の 内 容	対 策
FC error (illegal Function Call)	① マニュアルコマンドをプログラムで実行しようとした．概当するコマンドを次に示す． **CONT, PASS, RUN, EDIT, DELETE** ② プログラムコマンドをマニュアルで実行しようとした．概当するコマンドを次に示す． **END, LET, REM, STOP, LOCATE, DRAW, DRAW C, GOTO, GOSUB, RETURN, INPUT, DATA, READ, RESTORE, FOR～NEXT, IF～THEN ～ELSE～** ③ 継続できない状態で**CONT**コマンドを使用した．	① プログラム中から該当するコマンドを除き，再構成． ② 行番号をつけて実行させる． ③ BRKキーを押した後，最初から実行をやり直すか，**STOP**した行がわかっていれば**RUN**行番号でその次の行から実行を再開する．
FO error (next without FOr)	① **NEXT**文に対する**FOR**文がない．	① ネスティング構造を点検する
GS error (return without GoSub)	① **GOSUB**に対応しない**RETURN**文があった．	① **GOSUB**文におけるネスティング構造を確認する．メインルーチンとサブルーチンを明確にする．
MA error (MAthematical error)	① 数値演算あるいは数値関数演算で，その演算が不定あるいは不能である場合． 例）割り算で，0で割った場合．	① エラーが発生した行の数式を確認する．関連する変数の値を確認する．
NO error (Nesting Over error)	① ネスティングレベルが規定値を越えた． 例）**GOSUB～RETURN** 最大12レベル **FOR～NEXT** 最大6レベル	① ネスティング構造を確認し，レベル内に収める．

エラーメッセージ	エ ラ ー の 内 容	対 策
NR error (device Not Ready)	① I/Oデバイスが正しく接続されていない. 例) MTが接続されていない	① 関連機器のスイッチが正しくONされているか調べる. 正確に接続されているか調べる.
OM error (Out of Memory)	① RAMメモリの容量不足が発生した. 例) ●メモリ不足で配列宣言をした.	① 不要なプログラムを消去する. 拡張RAMパックでメモリ容量を増やす. SYSTEMコマンドで残バイト数を確認する.
OV error (OVer flow error)	① 演算結果または入力数値が10^{100}以上である.	① エラー発生行の数式を確認する. PRINT文をプログラム中に挿入し, その変数の値を確認する.
PR error (PRotected error)	① パスワード付きのプログラムに対して使用できないコマンドを実行しようとした. 概当するコマンドを次に表す. DELETE, LIST, LLIST, NEW, EDIT, ② パスワード付きプログラムに対して, 新たな行の追加, 削除をしようとした. ③ 異なるパスワードを入力した. ④ 本体と異なるパスワードのプログラムをLOADしようとした.	① ② 再度パスワードを入力し, ロックの状態を解除してから実行する. ③ 正しいパスワードを入力する. ④ 本体のパスワードを解除してからLOADする. その場合, LOADしたパスワードが本体の新たなパスワードになる.
RW error (Read Write error)	① LOAD, VERIFYコマンドを実行中にパリティエラーが発生した.	① 再度SAVEをし直す.
SN error (SyNtax error)	① コマンドの書式に誤りがある. ② 行番号に小数を含んだ場合. ③ 3次元以上の配列を宣言した場合.	① EDITでエラーが発生した行を呼び出し, 修正する. ② 行番号を修正する. ③ 2次元以内に修正する.

エラーメッセージ	エ ラ ー の 内 容	対 策
SO error (Stack Over flow)	① 数値スタックが8レベルを越えた. ② 演算子スタックが20レベルを越えた. ③ 文字スタックが10レベルを越えた.	① ② スタックがレベル内に収まるよう数式を簡略化, または分割する. ③ 文字式を簡略化または分割してスタックをレベル内に収める.
ST error (STring error)	① 許容文字変数長を越えた文字列を文字変数に代入しようとした. 次に許容文字変数長を示す. 固定文字変数 最大7文字 登録文字変数 最大16文字 文字配列変数 指定により最大79文字	① 文字長の多い変数に変更する. 代入すべき文字列の文字長を短かくする. 文字列を連結する場合, 注意が必要.
TM error (Type Missmatch)	① 代入文において左辺と右辺の変数型が異なる. ② 代入時の引数型が一致しない.	① 代入文の左辺と右辺を, 数値変数か文字変数で統一する. ② 数値変数には数値を, 文字変数には文字列を代入する.
UL error (Undefined Line number)	① 指定された行番号がない. **IF~THEN, GOTO, GOSUB** ② **GOTO**文, **GOSUB**文で指定されたプログラムエリアにプログラムがない.	① 分岐先行番号をつくるか, 分岐先行番号の指定を変更する. ② 分岐先のプログラムエリアをつくるか, 分岐先を変更する.
UV error (Undefined Variable)	① 定義されていない変数を用いた. ② **DIM**文宣言なしで配列変数を用いた.	① 使用する変数の初期値を確認する. ② プログラムの始めに, **DIM**文により配列を宣言する.
VA error (VAriable error)	① 41個以上の変数を登録しようとした.	① 登録変数と配列変数は合計40個まで使用できる. **LIST V**で変数名を確認し, **CLEAR**, **ERASE**で不要のものを削除する.

キャラクタコード表

コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字	コード	文字
0		32	(SPC)	64	@	96	`	128		160	(SPC)	192	タ	224	═
1		33	!	65	A	97	a	129	▁	161	。	193	チ	225	╞
2		34	"	66	B	98	b	130	▂	162	「	194	ツ	226	╪
3		35	#	67	C	99	c	131	▃	163	」	195	テ	227	╡
4		36	$	68	D	100	d	132	▄	164	、	196	ト	228	◢
5		37	%	69	E	101	e	133	▅	165	・	197	ナ	229	◣
6		38	&	70	F	102	f	134	▆	166	ヲ	198	ニ	230	◥
7		39	'	71	G	103	g	135	▇	167	ァ	199	ヌ	231	◤
8		40	(	72	H	104	h	136	▏	168	ィ	200	ネ	232	♠
9		41	)	73	I	105	i	137	▎	169	ゥ	201	ノ	233	♥
10		42	*	74	J	106	j	138	▍	170	ェ	202	ハ	234	♦
11	*(HOME)	43	+	75	K	107	k	139	▌	171	ォ	203	ヒ	235	♣
12	*(CLS)	44	,	76	L	108	l	140	▋	172	ャ	204	フ	236	●
13	*(RET)	45	−	77	M	109	m	141	▊	173	ュ	205	ヘ	237	○
14		46	.	78	N	110	n	142	▉	174	ョ	206	ホ	238	／
15		47	／	79	O	111	o	143	┼	175	ッ	207	マ	239	＼
16		48	0	80	P	112	p	144	┴	176	ー	208	ミ	240	╳
17	*(DEL)	49	1	81	Q	113	q	145	┬	177	ア	209	ム	241	円
18	*(INS)	50	2	82	R	114	r	146	┤	178	イ	210	メ	242	年
19		51	3	83	S	115	s	147	├	179	ウ	211	モ	243	月
20		52	4	84	T	116	t	148	▔	180	エ	212	ヤ	244	日
21		53	5	85	U	117	u	149	─	181	オ	213	ユ	245	時
22	*(ANS)	54	6	86	V	118	v	150	│	182	カ	214	ヨ	246	分
23	*(ENT)	55	7	87	W	119	w	151	▕	183	キ	215	ラ	247	秒
24	*(▲)	56	8	88	X	120	x	152	┌	184	ク	216	リ	248	〒
25		57	9	89	Y	121	y	153	┐	185	ケ	217	ル	249	市
26		58	:	90	Z	122	z	154	└	186	コ	218	レ	250	区
27		59	;	91	[	123	{	155	┘	187	サ	219	ロ	251	町
28	*(CSR>)	60	<	92	¥	124	¦	156	╭	188	シ	220	ワ	252	村
29	*(CSR<)	61	=	93	]	125	}	157	╮	189	ス	221	ン	253	人
30	*(CSR^)	62	>	94	^	126	~	158	╰	190	セ	222	゛	254	▩
31	*(CSRv)	63	?	95	_	127		159	╯	191	ソ	223	゜	255	

＊(RET)は，改行の機能コードとして動作します．

さくいん

＊アルファベット順＊

五十音順

●監修
渡辺　茂　東京大学名誉教授．都立工科短期大学学長．日本マイコンクラブ会長．著作多数．
●執筆
長谷川　晃　アオイ・エンジニアリング㈱代表取締役
繁野鎮雄　神奈川県マイコンクラブ
日本マイコンクラブ神奈川支部代表

昭和58年8月25日初版第1刷発行
昭和61年2月21日初版第6刷発行

監修
渡辺　茂
著者
長谷川　晃，繁野鎮雄
検印廃止

発行者
片岡　巌
発行所
技術評論社
東京都千代田区九段南2-4-13
九段光ビル　〒102
TEL　03−262−9351（代）
振替口座　東京0−76098

印刷　加藤文明社
製本　村上製本
G0036-02686 F

カシオ計算機株式会社営業本部

東京都新宿区西新宿2—6　新宿住友ビル
(〒160)　☎03-347-4811(代表)

カシオ計算機サービスセンター

地区	電話	郵便番号	住所
旭川	0166-23-8580	〒070	旭川市七条通り8丁目
札幌	011-231-2343	〒060	札幌市中央区南一条西12丁目
釧路	0154-24-8575	〒085	釧路市光陽町6—7
青森	0177-22-7466	〒030	青森市勝田2—1—12
秋田	0188-63-7690	〒010	秋田市山王2—1—40
盛岡	0196-24-2502	〒020	盛岡市本町通り3—19—6
仙台	0222-27-1404	〒980	仙台市一番町2—3—32
山形	0236-42-8018	〒990	山形市あこや町3—12—9
郡山	0249-33-5172	〒963	福島県郡山市香久池2—16—6
宇都宮	0286-34-0395	〒320	宇都宮市西大寛2—1—3
前橋	0272-53-3000	〒371	前橋市元総社町92—5
水戸	0292-25-6985	〒310	水戸市中央1—2—20
埼玉	0486-66-8567	〒330	大宮市大成町1—1 8 1
千葉	0472-43-1751	〒280	千葉市登戸町2—2 7 6
東京	03- 862-4141	〒101	千代田区神田佐久間町2—23
東京	03- 583-4111	〒106	港区六本木2—3—6
城南	03- 787-3721	〒145	大田区上池台1—1—6
城西	03- 376-3221	〒160	新宿区西新宿4—2—18
多摩	0425-23-3531	〒190	立川市錦町3—2—25
横浜	045-211-0821	〒231	横浜市中区弁天通り6—85
平塚	0463-23-2611	〒254	平塚市新宿1 1 9 6
新潟	0252-41-4105	〒950	新潟市米山3—1—5
長野	0262-28-9360	〒380	長野市岡田町30—20
甲府	0552-37-6371	〒400	甲府市城東2—22—11
沼津	0559-22-8928	〒410	沼津市高島町8—11
静岡	0542-81-8085	〒420	静岡市西中原1—4—35
浜松	0534-64-1658	〒435	浜松市西塚町3 2 4
豊橋	0532-53-2515	〒440	豊橋市魚町5
名古屋	052-263-0454	〒460	名古屋市中区栄4—6—15
岐阜	0582-62-0145	〒500	岐阜市鷹見町8
三重	0592-27-5066	〒514	津市鳥居町1 9 1
富山	0764-22-2251	〒930	富山市白銀町2—1
金沢	0762-37-8511	〒920	金沢市諸江町下丁93—1
京都	075-351-1161	〒600	京都市下京区五条通り堀川東入ル
大阪	06 362-8181	〒530	大阪市北区南森町2—1—20
奈良	0742-24-3811	〒630	奈良市西木辻町2 0 0—6 2
和歌山	0734-31-7807	〒640	和歌山市九家の丁5
神戸	078-392-4123	〒650	神戸市中央区北長狭通り4—4—18
岡山	0862-41-8471	〒700	岡山市西古松4 0 6—2
松江	0852-25-1311	〒690	松江市嫁島町1 7 3
福山	0849-24-2830	〒720	福山市南本庄町2—1 3 0
広島	082-263-1090	〒730	広島市稲荷町4—1
山口	0835-22-6164	〒747	防府市戎町1—10—16
高松	0878-62-5240	〒760	高松市亀岡町9—16
松山	0899-45-2234	〒790	松山市平和通り1—1—5
福岡	092-411-2684	〒812	福岡市博多区博多駅南1—2—15
長崎	0958-61-8084	〒852	長崎市宝栄町2—26
熊本	0963-67-0650	〒862	熊本市健軍4—1—5
鹿児島	0992-56-3575	〒890	鹿児島市上荒田町30—18

規格

型　式	PB-700
基本計算機能	負数，指数，カッコを含む四則計算（加減・乗除の優先順位判別機能つき）　MOD
組込関数機能	三角・逆三角関数（角度単位は度・ラジアン・グラジアン），対数・指数関数，開平，べき乗，整数化，整数部除去，絶対値符号化，四捨五入，乱数，円周率

関数桁容量	入力範囲	答の精度
sin x, cos x, tan x	$\|x\|<5400$（30 πrad，6000 gra）	基本的に10桁目±1
$\sin^{-1}x$, $\cos^{-1}x$	$\|x\|\leqq 1$	〃
$\tan^{-1}x$	$\|x\|<10^{100}$	〃
log x, ln x	$x>0$	〃
e^x	$x\leqq 230$	〃
$\sqrt{x}$	$x\geqq 0$	〃
$x^y(x\uparrow y)$	$x>0$	〃

注　ただし，tan Xにおいては，以下の場合は除きます．
DEG：$|x|=90(2n-1)$
RAD：$|x|=\frac{\pi}{2}(2n-1)$
GRAD：$|x|=100(2n-1)$
（nは整数）

コマンド	CONT, DELETE, EDIT, LIST, LLIST, LOAD, NEW, PASS, PROG, RUN, SAVE, SYSTEM, VERIFY, ANGLE, BEEP, CHAIN, CLEAR, CLS, DATA, DIM, DRAW, DRAWC, END, ERASE, FOR - TO - STEP, NEXT, GET, GOSUB, GOTO, IF-THEN-ELSE, INPUT, LET, LOCATE, LPRINT, PRINT, PUT, READ, REM, RESTORE, RETURN, STOP, TROFF, TRON
プログラム関数	ASC, CHR$, VAL, STR$, LEFT$, RIGHT$, MID$, LEN, INKEY$, TAB, USING, POINT, SIN, COS, TAN, ASN, ACS, ATN, SQR, LOG, LGT, EXP, ABS, INT, FRAC, SGN, ROUND, PI, RND
計算範囲	$\pm 1\times 10^{-99}\sim\pm 9.999999999\times 10^{99}$（内部演算は仮数部12桁を使用，精度は10桁目±1）
プログラム言語	BASIC（ベーシック）
プログラム容量	本体実装4kByte，最大16kByteまで拡張可　RAM　標準4kByte，最大16kByte　ROM　約25kByte
組込プログラム数	最大10組（P0～P9）

スタック数	サブルーチン　12段 FOR-NEXTループ　6段 数　値　8段 演算子　20段
表示方式	仮数部10桁＋指数部2桁
表示素子	32×160ドット液晶　(4×20キャラクター)
主要素子	C-MOS LSI　他
電源	単3電池4本, CR-1220 (バックアップ用) 1個使用
消費電力	約　10mA
電池寿命	AM-3 (アルカリマンガン電池)　約120時間 SUM-3 (本体装着電池)　約80～100時間 バックアップ用 (単3電池装着時)　CR-1220　約2年間
オートパワーオフ	約　8分
使用温度	0℃～40℃
大きさ・重さ	幅200 奥行88 高さ23㎜, 重さ315g
付属品	ソフトケース,「PB-700活用法」, コマンドガイド
別売品	CM-1 専用マイクロカセットテープレコーダー OR-4 オプションRAM (4KB) FA-10　カセットインタフェイス付ミニプロッタプリンタ